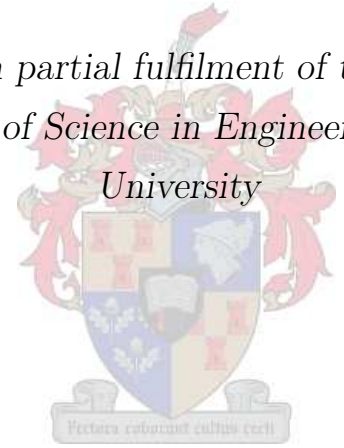
# Unsupervised clustering of audio data for acoustic modelling in automatic speech recognition systems.

by

George Willem Goussard

*Thesis presented in partial fulfilment of the requirements for the degree Master of Science in Engineering at Stellenbosch University*



Supervisor: Prof. T.R. Niesler

Department of Electrical and Electronic Engineering

March 2011

# Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Signature: .....

G.W. Goussard

Date: 2011/02/09 .....

# Abstract

This thesis presents a system that is designed to replace the manual process of generating a pronunciation dictionary for use in automatic speech recognition. The proposed system has several stages.

The first stage segments the audio into what will be known as the sub-word units, using a frequency domain method. In the second stage, dynamic time warping is used to determine the similarity between the segments of each possible pair of these acoustic segments. These similarities are used to cluster similar acoustic segments into acoustic clusters. The final stage derives a pronunciation dictionary from the orthography of the training data and corresponding sequence of acoustic clusters. This process begins with an initial mapping between words and their sequence of clusters, established by Viterbi alignment with the orthographic transcription. The dictionary is refined iteratively by pruning redundant mappings, hidden Markov model estimation and Viterbi re-alignment in each iteration.

This approach is evaluated experimentally by applying it to two subsets of the TIMIT corpus. It is found that, when test words are repeated often in the training material, the approach leads to a system whose accuracy is almost as good as one trained using the phonetic transcriptions. When test words are not repeated often in the training set, the proposed approach leads to better results than those achieved using the phonetic transcriptions, although the recognition is poor overall in this case.

# Opsomming

Die doelwit van die tesis is om 'n stelsel te beskryf wat ontwerp is om die handgedrewe proses in die samestelling van 'n woordeboek, vir die gebruik in outomatiese spraakherkenningsstelsels, te vervang. Die voorgestelde stelsel bestaan uit 'n aantal stappe.

Die eerste stap is die segmentering van die oudio in sogenaamde sub-woord eenhede deur gebruik te maak van 'n frekwensie gebied tegniek. Met die tweede stap word die dinamiese tydverplasingsalgoritme ingespan om die ooreenkoms tussen die segmente van elkeen van die moontlike pare van die akoestiese segmente bepaal. Die ooreenkomste word dan gebruik om die akoestiese segmente te groepeer in akoestiese groepe. Die laaste stap stel die woordeboek saam deur gebruik te maak van die ortografiese transkripsie van afrigtingsdata en die ooreenstemmende reeks akoestiese groepe. Die finale stap begin met 'n aanvanklike afbeelding vanaf woorde tot hul reeks groep identifiseerders, bewerkstellig deur Viterbi belyning en die ortografiese transkripsie. Die woordeboek word iteratief verfyn deur oortollige afbeeldings te snoei, verskuilde Markov modelle af te rig en deur Viterbi belyning te gebruik in elke iterasie.

Die benadering is getoets deur dit eksperimenteel te evalueer op twee subversamelings data vanuit die TIMIT korpus. Daar is bevind dat, wanneer woorde herhaal word in die afrigtingsdata, die stelsel se benadering die akkuraatheid ewenaar van 'n stelsel wat met die fonetiese transkripsie afgerig is. As die woorde nie herhaal word in die afrigtingsdata nie, is die akkuraatheid van die stelsel se benadering beter as wanneer die stelsel afgerig word met die fonetiese transkripsie, alhoewel die akkuraatheid in die algemeen swak is.

# Acknowledgements

I would like to express my sincere gratitude to the following people who have contributed to making this work possible:

- Prof. Thomas Niesler of the University of Stellenbosch as my supervisor.

# Dedications

*This thesis is dedicated to my wife,*
*Heidi Weyers,*
*for her support, patience and love.*

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

This thesis addresses the automatic generation of acoustic units and an associated pronunciation dictionary for use in speech recognition. Traditionally, acoustic units and pronunciations are determined by a professional linguist. Usually phonemes form the acoustic units and the dictionary is a mapping between the words and their respective sequences of phonemes. To train a speech recogniser, a professional linguist must therefore provide pronunciations for all the words in the audio training material. This allows the speech recogniser to build models of the relevant phonemes referenced in the dictionary. However, this procedure is extremely cumbersome and expensive. Figure 1.1 compares the traditional process of dictionary generation with the solution proposed in this thesis. The latter will attempt to automatically generate both the acoustic units and the dictionary, using only the audio and the orthographic transcription as input. If successful, this would increase the speed and reduce the cost of developing a speech recogniser for cases in which linguist-produced pronunciation dictionaries are not available. This is the case, for example, in many South African languages and accents.

## 1.1 Directly related work

In reviewing the literature, only a single paper was found where the authors had attempted to train a speech recognition system using automatically determined subword units. In the work by Singh *et al.* (2002) an iterative approach is proposed in which a set of acoustic models and an automatically generated pronunciation dictionary are incrementally updated, as shown in Figure 1.2.

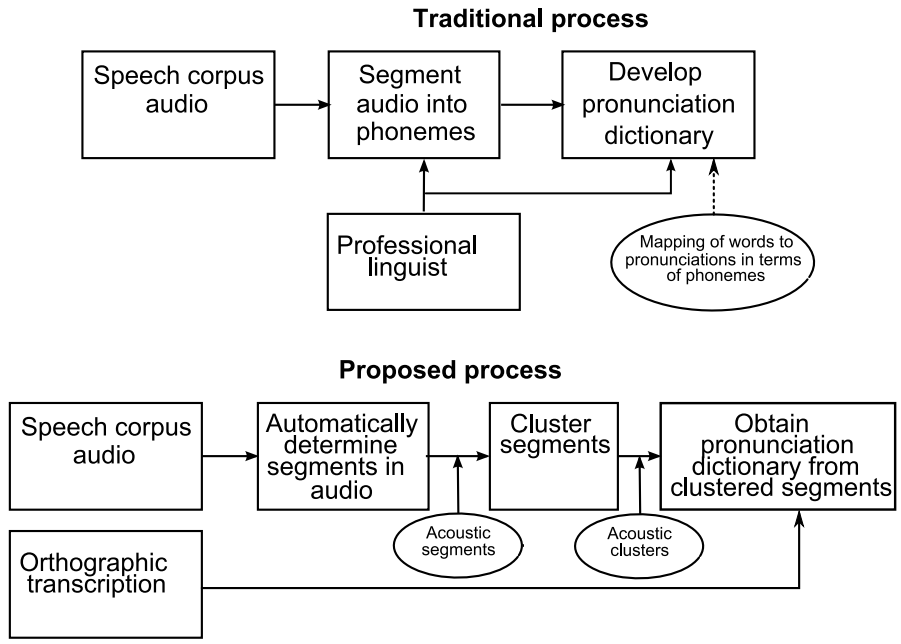



**Figure 1.1:** Comparison of traditional and proposed approaches to pronunciation dictionary generation.

The system is bootstrapped from a configuration in which the graphemes of the words in the dictionary are used as subword units, and hence the alphabet used by the system defines the number of subword units. The approach proposed by the author will not make this assumption, but will attempt to find subword units by direct inspection of the audio training data. Nevertheless, the iterative approach proposed by Singh *et al.* (2002) has influenced the design of the system proposed in this thesis.

## 1.2 Development environment

Initial development of the algorithms described in this thesis was carried out on an Ubuntu Linux 32-bit computer workstation. Full-scale experiments were later executed on the high performance Linux computing cluster at the Department of Mechanical Engineering. Algorithms were implemented in C and interconnected using BASH shell scripts.

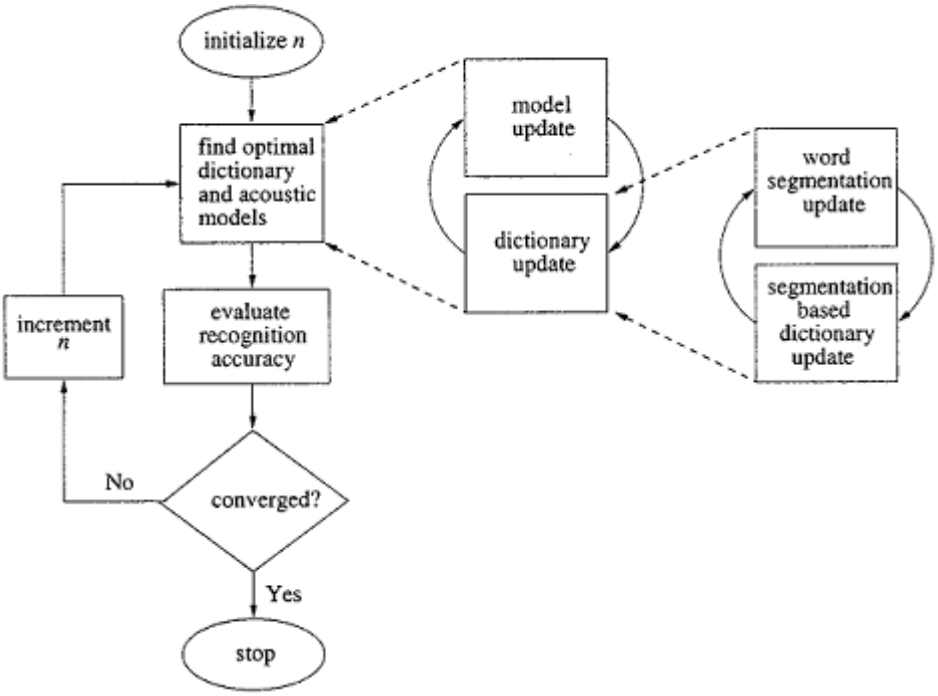



**Figure 1.2:** Flow diagram of the procedure described in Singh *et al.* (2002).

## 1.3 Thesis structure

The approach proposed in this thesis can be divided into stages, as indicated in Figure 1.1. The author will assume that only the acoustic data and orthographic transcription are available as input. The first stage is responsible for segmenting the audio into acoustic segments. Some acoustic segments may sound similar, and one would like a collective label for such sounds. Hence the second stage identifies similar segments by means of a clustering procedure. The third and final stage generates a mapping between the words in the orthographic transcription and these acoustic clusters. The result is a segmentation of the audio into "phoneme-like" acoustic units as well as a corresponding automatically generated pronunciation dictionary.

The next chapter will briefly describe the audio data used to develop and test the system. Chapters 3, 4 and 5 then describe the three stages discussed above in turn.

# Chapter 2

# Data

## 2.1 Overview

The methods proposed in this thesis were tested using the TIMIT speech corpus. This is a well-known speech corpus used in the development of speech processing and recognition systems. The corpus was recorded at Texas Instruments (TI) at a sample rate of 16kHz, transcribed at the Massachusettes Institute of Technology (MIT) and is maintained by the American National Institute of Standards and technology (NIST). A brief description of the corpus will be presented next.

**Table 2.1:** Dialect distribution of speakers in the TIMIT speech corpus. Percentages in the "Male" and "Female" columns indicate proportions for the dialect in question. Percentages in the "Total" column are of the total number of speakers.

| Regions | Male | Female | Total |
|---|---|---|---|
| New England | 31 (63%) | 18 (27%) | 49 (8%) |
| Northern | 71 (70%) | 31 (30%) | 102 (16%) |
| North Midland | 79 (67%) | 23 (23%) | 102 (16%) |
| South Midland | 69 (69%) | 31 (31%) | 100 (16%) |
| Southern | 62 (63%) | 36 (37%) | 98 (16%) |
| New York City | 30 (65%) | 16 (35%) | 46 (7%) |
| Western | 74 (74%) | 26 (26%) | 100 (16%) |
| Army brat | 22 (67%) | 11 (33%) | 33 (5%) |
| Total | 438 (70%) | 192 (30%) | 630 (100%) |

## 2.2 Brief description of the TIMIT corpus

The TIMIT speech corpus contains a total of 6300 recorded sentences collected from eight different dialect regions in the United States as listed in Table 2.1. A total of 10 sentences were recorded for each speaker, and there are male and female speakers from each dialect region. The eighth dialect region ("army brat") is not fixed geographically, but comprises speakers that moved around a lot during their childhood.

Audio files can be distinguished by the prefixes in their filenames and the recorded sentences can be divided into three categories:

- Phonetically diverse sentences (SI). These sentences were chosen from a large corpus of existing text to provide rich phonetic coverage and were designed to exploit the differences in the dialects. The audio filenames starting with SI are the phonetically diverse sentences.

- Phonetically compact sentences (SX). These sentences were designed by hand to provide a rich variety of phonetic segments and phonetic contexts. The audio filenames starting with SX are the phonetically compact sentences.

- Unique sentences (SA). These two sentences were specially designed to demonstrate the effect of dialect on the acoustic characteristics of American English speech. The audio filenames starting with SA indicate the unique sentences.

**Table 2.2:** Description of the TIMIT corpus.

| Sentence type | Sentences | Speakers | Total | Sentences/Speaker |
|---|---|---|---|---|
| Dialect (SA) | 2 | 630 | 1260 | 2 |
| Compact (SX) | 450 | 7 | 3150 | 5 |
| Diverse (SI) | 1890 | 1 | 1890 | 3 |
| Total | 2342 | | 6300 | 10 |

Table 2.2 presents a breakdown of the TIMIT speech corpus composition. According to Table 2.2, the only sentences repeated by every speaker are the SA sentences. In addition each speaker reads five phonetically compact (SX) sentences, and each SX sentence is read by seven different speakers. Finally,

each speaker also reads three phonetically diverse (SI) sentences, each SI sentence being read by only one speaker.

## 2.3 TIMIT subsets used for experimentation

In this thesis, two subsets of the TIMIT corpus are used for experimental evaluation, as shown in Table 2.3. They are:

- The SA sentences are used in isolation for testing. Due to the very small closed vocabulary and the high repetition rate of each word in the training data, this subset should provide a very optimistic scenario for the automatic subword unit determination methods proposed in this thesis.
- A subset of the SX and SI sentences was chosen to ensure that all words in the respective test set are also present in the training set (i.e. a closed vocabularly). Test set words are repeated with varying frequency in the training set. The SI+SX subset is a more realistic scenario in which to evaluate the automatic baseform determination methods proposed in this thesis.

**Table 2.3:** TIMIT subsets used for experimentation.

| | TIMIT subsets | | | |
|---|---|---|---|---|
| | SA | | SI+SX | |
| | Train | Test | Train | Test |
| Vocabulary size (word types) | 21 | 21 | 2602 | 311 |
| Number of utterances | 924 | 48 | 1307 | 129 |
| Number of speakers | 462 | 24 | 450 | 93 |
| Duration (minutes) | 47.8 | 2.4 | 69.5 | 5.2 |

Table 2.3 summarises the SA and SI+SX subsets used in the experimental evaluation of the system proposed by this. The duration of the SI+SX subset also influenced the selection. Computing time significantly increases the longer the duration of the training data.

## 2.4 Summary and conclusion

This chapter has presented a brief overview of the TIMIT speech corpus. Two subsets of this corpus have been identified for experimental evaluation of the algorithms used by the system proposed by this thesis. The next chapters will introduce these algorithms and provide the experimental results.

# Chapter 3

# Segmentation

## 3.1 Overview

In order to train a speech recogniser, a mapping between words and their respective sequences of phonemes is needed. This mapping is generally referred to as the pronunciation dictionary. The training process then uses these pronunciations in conjunction with the orthographic transcriptions and the audio data, to build statistical models of the phonemes. In this thesis the author assumes to have at his disposal the audio data of the spoken utterances and the accompanying orthographic transcription, but not the pronunciations. This means that one needs to somehow automatically segment the audio into suitable subword units, and then use these in conjunction with the orthography to obtain a pronunciation dictionary. The first step in this process is the segmentation of the audio into "phoneme-like" units, and is the focus of this chapter.

The segmentation stage is extremely important, because every subsequent stage will rely on the quality of these segments. One important attribute of these acoustic segments is their size. If the acoustic segment is too short then it may have little meaning, and if it is too long then it may be composed of more than one meaningful acoustic unit. A benchmark that will be used to evaluate this stage is a comparison of the automatically determined segments with the manually derived phonetic segments. The TIMIT speech corpus, which was used to perform the experiments in this thesis, is defined by forty-eight phonemes. A good segmentation algorithm might be considered to be segmention of the speech into forty-eight comparable segments. Finding a way

to automatically segment audio is not a trivial task, and one needs to start by considering the relevant literature.

## 3.2 Literature review

Although the development of a complete speech recognition system using automatically derived subword units has been considered only by Singh *et al.* (2002), as discussed in Chapter 1, a number of authors have considered the segmentation of audio data. Their work is surveyed in the following sections.

### 3.2.1 Voiced, unvoiced and silence segmentation

The broadest class of segmentation algorithms was found to consider the classification of the speech signal into voiced and unvoiced regions. Knowing where voicing occurs is important in some speech coding algorithms (Tolba and O'Shaugnessy, 1998; Ahmadi and Spanias, 1999). It has also been used to improve speech recognition performance (Zolnay *et al.*, 2003). These approaches often work by extracting time domain features, such as zero-crossing rate (Ahmadi and Spanias, 1999), energy (Arcienega and Drygajlo, 2002; Ahmadi and Spanias, 1999; Loukina *et al.*, 2009), and correlation (Arcienega and Drygajlo, 2002; Zolnay *et al.*, 2003; Tolba and O'Shaugnessy, 1998). In some cases, frequency domain features such as MFCCs (Ahmadi and Spanias, 1999) and harmonic product spectrum (Zolnay *et al.*, 2003) have also been shown to be useful.

Many of the algorithms studied required significant tuning by the implementor, and included a number of parameters that need to be optimised before use. In some cases, a small subset of user labelled data was also required, in which case the method cannot be considered truly unsupervised.

During preliminary tests, segmentation based on voicing seemed not to be a good basis for discovering subword units, since regions of sustained voicing led to very long segments.

### 3.2.2 Techniques based on automatic keyword discovery

A different family of algorithms tries to identify recurring phrases in unlabelled audio. These techniques are based on an alternative implementation of

the dynamic time warping (DTW) algorithm, which allows it to detect local sub-matches between two audio segments (Park and Glass, 2005, 2008; Gajjar *et al.*, 2008; Muscariello *et al.*, 2009). These techniques are particularly suited to the detection of frequently recurring words or phrases in unlabelled audio from a single speaker and within a stable acoustic environment. However, these techniques do not attempt to segment all the audio, but only to find frequently recurring sub-portions. Furthermore, no use is made of the orthographic transcriptions. In fact it is assumed that these are not available.

### 3.2.3 Techniques based on detection of transients in audio

A few researchers have considered approaches that attempt to find segment boundaries in unlabelled audio by detecting points at which the time or spectral characteristics of the speech signal change strongly.

To the best knowledge of the author, the work by Aversano *et al.* (2001) was the first to consider this approach. In their work, significant discontinuities in the main spectra were identified. The number of frequency bands taken into account was determined by a critical-band analysis. The technique suffers from the requirement of three parameters that need to be specified before use.

An alternative approach that attempts to achieve the same goals was presented a few years later by Murthy and Gadde (2003) and by Golipour and O'Shaughnessy (2007). Here the authors make use of the self-proposed "modified group delay" function, to detect transitions between consecutive segments in unlabelled audio. However, their technique also requires several parameters to be optimised and was difficult to implement. Despite being given access to the source code obtained from the author, results could not be reproduced.

Finally, ten Bosch and Cranen (2007) proposed an approach that may be seen as a substantial refinement on the work by Aversano *et al.* (2001), leading to a more elegant algorithm requiring only a single user parameter (ten Bosch *et al.*, 2006; ten Bosch and Cranen, 2007). This approach will be described in greater detail in Section 3.3.

### 3.2.4 Other relevant techniques

Some research in the field of language identification has also considered the problem of dividing the audio speech data into automatically generated segments. In Torres-Carrasquillo *et al.* (2002) a Gaussian mixture model (GMM) is used to cluster the acoustic vectors of the training material as a first step. Then, during operation, the Gaussian mixture component to which each test feature vector is assigned with the highest probability, is considered to be the segment label and passed to a statistical language model for further processing. This process is referred to as "GMM tokenisation". Although this approach is completely unsupervised, it does not, usually, segment a speech signal into appropriately sized segments. Each input feature vector can, normally, be assigned to a different mixture component, and hence this segmentation does, in general, not result in "phoneme-like" units.

## 3.3 Audio segmentation as proposed by ten Bosch

After initial informal evaluation of some of the segmentation methods reviewed in the previous section, the author has chosen the method described by ten Bosch and Cranen (2007) for integration into the proposed approach. The following sections will describe and evaluate this method in greater detail.

### 3.3.1 Pre-processing

The first step is to generate Mel-frequency cepstral coefficient (MFCC) vectors from the audio. The MFCC coefficients are a parameterization of the time domain samples of the original speech and are a popular choice of feature vectors in automatic speech recognition. Various parameters have to be considered, for example, the rate at which these feature vectors are generated and the number of MFCC coefficients that are calculated. The rate at which the MFCC vectors are generated is related to the size of the frames used to analyse the audio signal and the overlap between successive frames. A small frame length with no overlap can generate just as many feature vectors as a large frame length, but with large overlap. The number of MFCC coefficients is important, because the more coefficients there are to describe a frame, the

more accurately the feature vector describes the speech in the frame. As discussed by ten Bosch and Cranen (2007), the number of MFCC coefficients chosen was twelve, with the addition of log energy, delta and delta-delta coefficients. This results in an MFCC vector size of thirty-nine coefficients (13x3). MFCC vectors are generated at a rate of one every 10msec (160 samples) and the window size is 20msec (320 samples). These specifications correspond to a half-frame overlap.

### 3.3.2 Distance measure

After the calculation of MFCC vectors, a distance measure between consecutive feature vectors is defined in order to detect points at which the speech signal changes rapidly, and hence a segment boundary might be considered. The following measure is considered by ten Bosch and Cranen (2007):

$$d(v_1, v_2) = \arccos\left(\frac{v_1^t \cdot v_2}{\sqrt{(v_1^t \cdot v_1)\,(v_2^t \cdot v_2)}}\right)$$

where $v_1$ and $v_2$ are any two consecutive MFCC vectors. The definition of the dot product between two consecutive vectors $v_1$ and $v_2$ is:

$$v_1^t \cdot v_2 = \|v_1\|\,\|v_2\| \cos\theta$$

This distance measure therefore corresponds to the angle between two consecutive vectors. The angle has the unit of radians and is not dependent on the vector magnitudes. If the vectors $v_1$ and $v_2$ are similar, the angle between them is small and thus $d(v_1, v_2)$ will be small. If $v_1$ and $v_2$ point in opposite directions, however, $d(v_1, v_2)$ will reach its maximum angle of $\pi$.

### 3.3.3 Segmentation criterion

The distance measure described in the previous section is used to segment the stream of MFCC vectors by means of the following criterion:

$$D(i) = \log(E(v_i)) \cdot d\left(\left(v_{(i-2)} + v_{(i-1)}\right)/2, \left(v_{(i+1)} + v_{(i+2)}\right)/2\right) > \delta \qquad (3.3.1)$$

This criterion states that the angle between two MFCC vectors will be weighted by the log energy of the current frame in order to make a segmentation

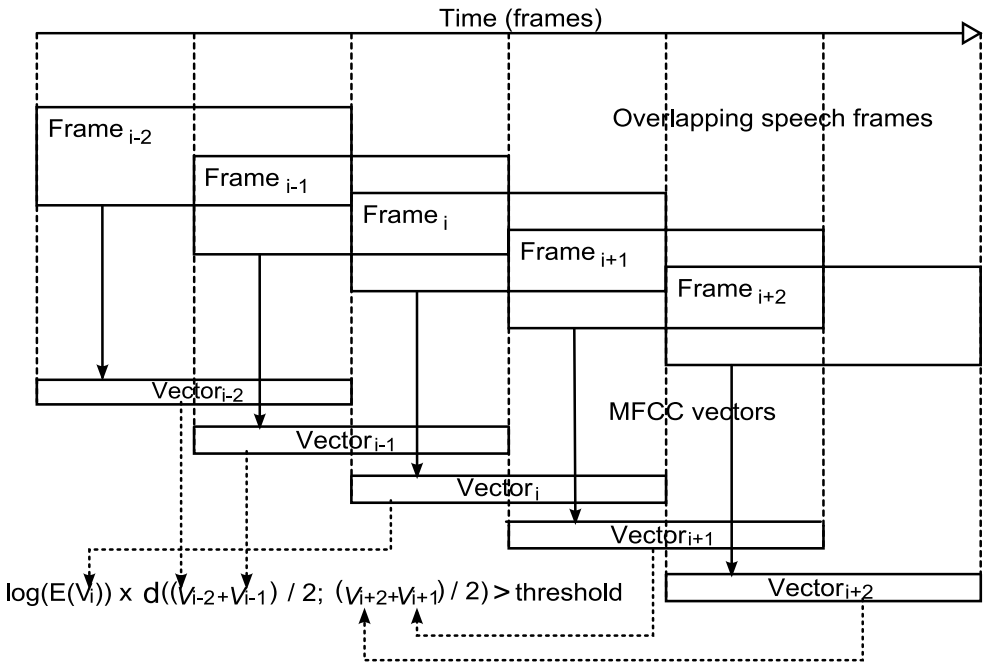



**Figure 3.1:** Calculation of the segmentation decision criterion.

decision. Furthermore, the angle is calculated between the average of the two MFCC vectors preceeding the current frame, and the average of the two following it. The MFCC vector corresponding to the current frame in equation 3.3.1 is $v_i$. The averages are used in order to take into account the variation of angle over a number of speech frames and not just two successive vectors. Transitions in the speech signal generally take place over a few frames and not just abruptly between two frames. Figure 3.1 illustrates the calculation of the decision criterion $D(i)$.

By weighting the angle between MFCC vectors with $\log(E(v_i))$, the log energy of the current MFCC vector $v_i$, the segmentation criterion is made dependent on the energy in the current frame and thereby prevents the insertion of segments in low energy regions. Silence regions have low energy but are not guaranteed to have low variation in angle. Hence $D(i)$ will emphasise regions with sharp changes in speech charateristics (high angle variation) and speech with high energy. The user specified threshold value can now be any value, since the log energy $\log(E(v_i))$ of the current MFCC vector $v_i$ is generally not bounded. To circumvent this problem, $D(i)$ is normalised by the maximum energy in the utterance.

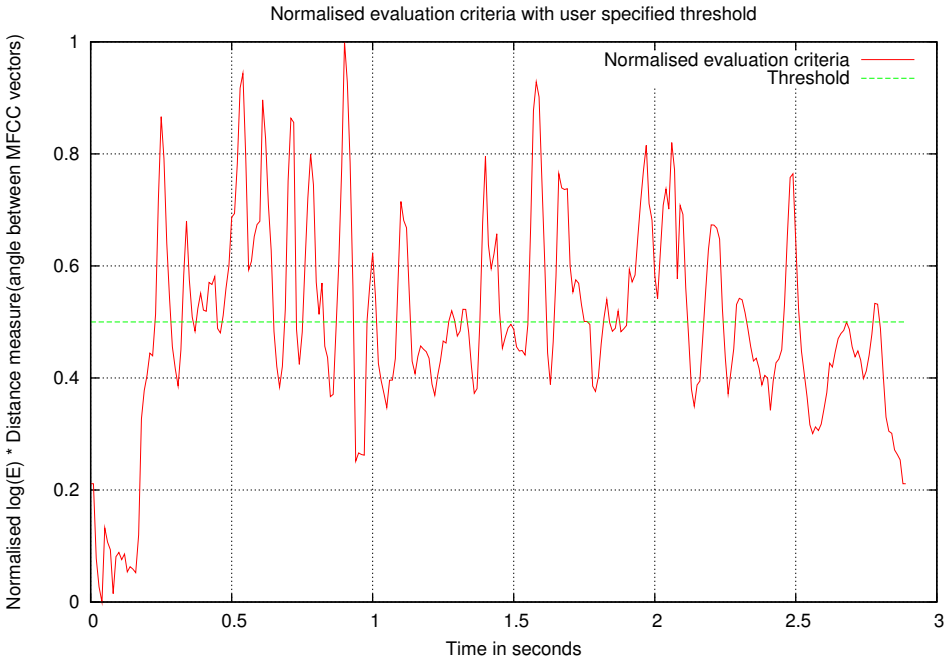



**Figure 3.2:** Normalised segmentation criteria $D(i)$ over time for the TIMIT utterance: "She had your dark suit in greasy wash water all year". Threshold indicated at a value of 0.5.

Following the normalisation, the user can set the threshold to a value between zero and one. Figure 3.2 shows a plot of the normalised $D(i)$ over time for an utterance. The next section will describe the algorithm proposed by ten Bosch and Cranen (2007) used to identify segment boundaries from the normalised distance measure.

### 3.3.4 Segmentation

The audio is segmented by searching for all the peaks in the normalised distance measure above the threshold set by the user. A peak is defined as a value of $D(i)$ which is higher than the preceeding and the succeeding value. Initially, each peak is considered a segment boundary in the audio. The first acoustic segment stretches from the start of the utterance until the first peak. The last acoustic segment stretches from the last peak until the end of the utterance. From the example illustrated in Figure 3.2, it is evident that the normalised segmentation criterion has many smaller peaks. If all of the peaks are considered to be acoustic segments, this would result in many very short

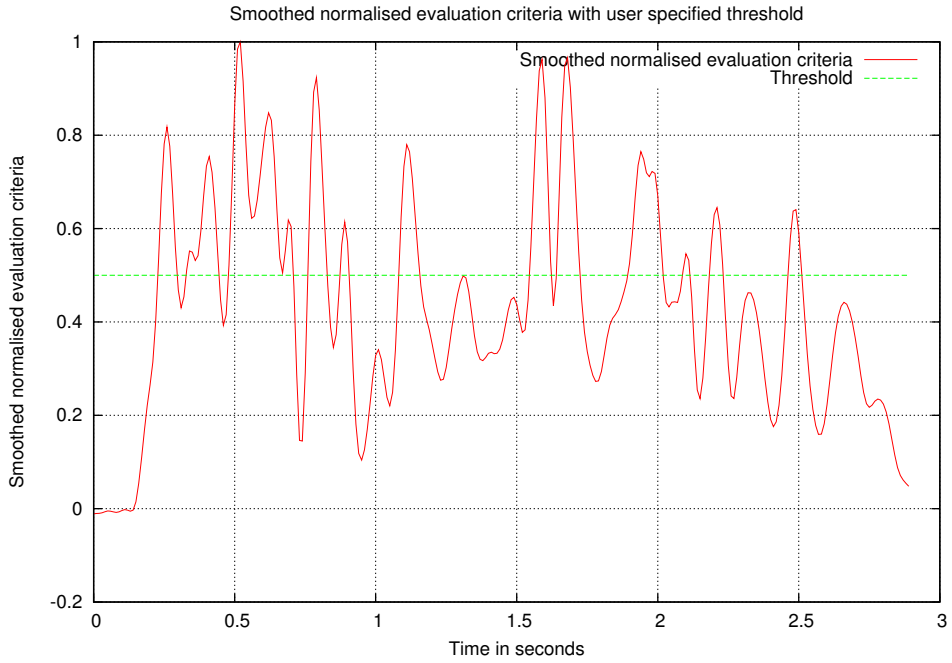



**Figure 3.3:** Smoothed normalised evaluation criteria over time: "She had your dark suit in greasy wash water all year". Threshold indicated at a value of 0.5.

acoustic segments. To overcome this, $D(i)$ can either be smoothed, or the peak detection algorithm can be refined. It was decided to smooth the segmentation criterion using a nine-point Hanning window, as also proposed by ten Bosch and Cranen (2007). This window is arranged symmetrically around the current value of interest $D(i)$. In order to obtain smooth values for the first three and last four values of $D(i)$, it is assumed that $D(i) = D(0)$ for $i < 0$ and $D(i) = D(N-1)$ for $i \geq N$. Figure 3.3 illustrates the result of smoothing the $D(i)$ sequence shown in Figure 3.2.

In the implementation, the output of the segmentation stage is a text file in which the boundaries of the segments are given and the acoustic segments are indicated by numbered labels. An extract from such an output file is presented in Figure 3.4 with the corresponding file's spectrogram and waveform presented in figures 3.5 and 3.6 respectively. Some of these acoustic segments may sound similar, and these should be clustered together. The clustering stage, which takes the text file with the acoustic segment labels illustrated in Figure 3.4 as input, is the focus of the next chapter.

```
#!MLF#
<some_filename>
500000  4100000 R0
4100000  5200000 R1
5200000  6200000 R2
6200000  7000000 R3
7000000  7800000 R4
7800000  9000000 R5
9000000 11100000 R6
11100000 15800000 R7
15800000 16700000 R8
16700000 19500000 R9
19500000 19800000 R10
19800000 20900000 R11
20900000 22100000 R12
22100000 24800000 R13
.
<some_filename>
```

**Figure 3.4:** Extract of the output file of the segmentation stage. Segment boundaries are indicated in 100ns units, for compatibility with the HTK tools.

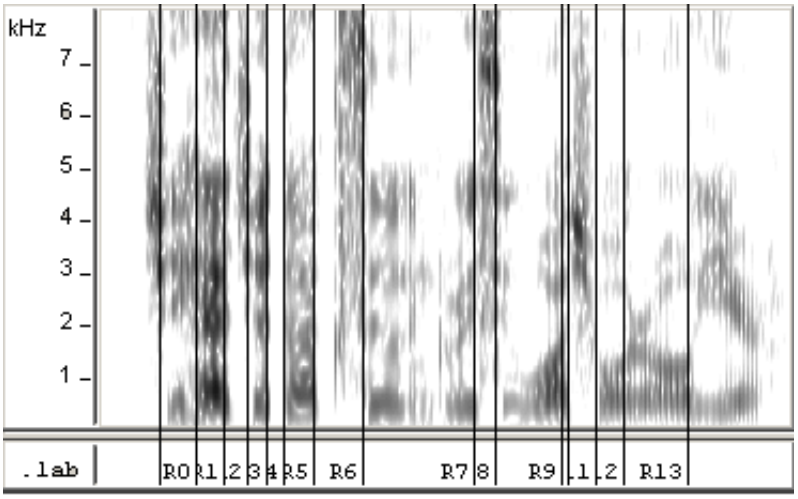



**Figure 3.5:** Spectrogram and region boundaries for the phrase: "She had your dark suit in greasy wash water all year".

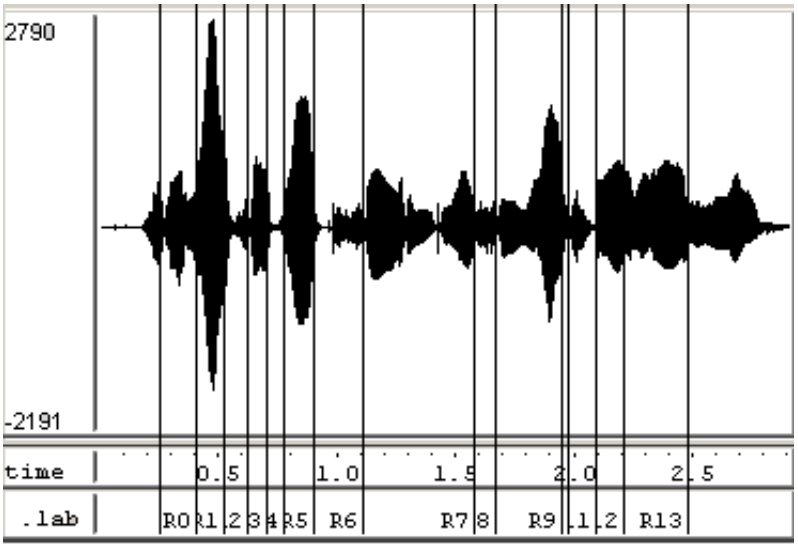



**Figure 3.6:** Waveform and region boundaries for the phrase: "She had your dark suit in greasy wash water all year".

## 3.4 Summary and conclusion

This chapter has reviewed methods that can be used to automatically segment audio data. The method suggested by ten Bosch and Cranen (2007) was selected, and described in greater detail. Finally, the operation of the algorithm was demonstrated for an example utterance.

# Chapter 4

# Clustering

## 4.1 Overview

This chapter deals with the clustering of acoustic segments according to their acoustic similarity. This set of clusters will be used in the following chapter to automatically generate a dictionary that consists of a mapping between words and their respective sequences of acoustic clusters. In order to cluster acoustic segments one needs a measure of how similar any two are, and to develop a procedure to cluster them accordingly.

In order to determine how similar two acoustic segments of unequal length are, the dynamic time warping (DTW) algorithm was used. The DTW algorithm can be considered an application of dynamic programming, where the goal is to find the optimal alignment between two sequences, given some constraints and costs involved in the decisions. The result of the DTW algorithm is a trellis which indicates the best frame-by-frame alignment between any two acoustic segments, as well as an overall score which can be used to quantify the quality of the alignment between those particular two segments. By means of DTW, these overall scores are calculated for each two acoustic segments that must be compared. The resulting similarities between all pairs of acoustic segments are arranged in a lower triangular matrix. Figure 4.1 illustrates the correspondence between the DTW trellis and this matrix. Each column and row of the matrix represents an acoustic segment. All entries on the diagonal are zero, since these represent the similarity of an acoustic segment with itself. The values in this matrix are then used to perform the clustering.

Clustering is the process of grouping individual units (in this case seg-

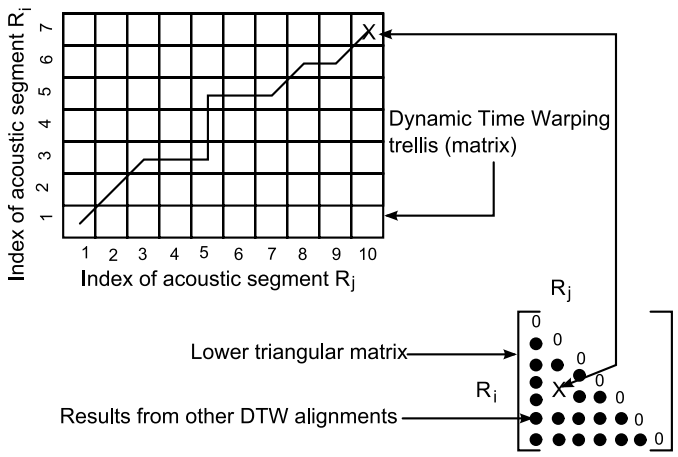



**Figure 4.1:** Correspondance between the DTW algorithm result and the lower triangular matrix of similarities. $R_i$ and $R_j$ are two arbitrary acoustic segments.

ments of speech) together, based on their similarity to each other. Clustering requires the computation of a distance (dissimilarity) between units (in this case by DTW) and is usually iterative, with some convergence criterion. If the clustering is supervised, it is usually referred to as classification. This means there are predefined models with labels assigned to them. Data is then classified according to the extent to which the data matches a model. Constructing the models might require some training data. However, in this thesis the emphasis is on unsupervised clustering, and how it is used to automatically find acoustic subword units for speech recognition. The clustering is unsupervised because there is no predefined set of labelled phonemes that can be used to classify acoustic segments of speech.

In the proposed method, only the similarity between acoustic segments is known. Unsupervised cluster methods like k-means algorithm cannot be applied here, since the acoustic segments themselves are not associated with vectors in some vector space. Instead, the author will make use of agglomerative hierarchical clustering, which requires only the similarities between the units to be clustered to be known (Everitt, 1993). Initially, all acoustic segments represent individual and separate clusters. They are merged successively in an iterative fashion. This is visualised in Figure 4.2, where hierarchical clustering approaches are represented as a tree hierarchy.

The remainder of this chapter is devoted to the details of the DTW algorithm and the cluster merging procedure.

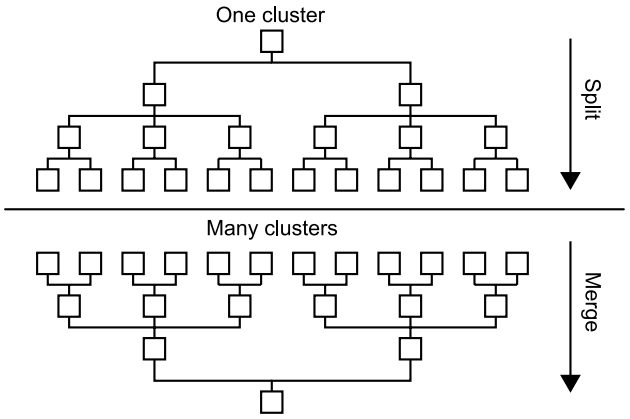



**Figure 4.2:** Different approaches towards hierarchical clustering data (dendograms). Agglomorative hierarchical clustering is illustrated below, while a divisive approach is illustrated above.

## 4.2 Similarity computation by DTW

Dynamic programming tries to solve a problem which usually involves a number of possible decisions with associated costs and constraints. The optimal solution is the sequence of decisions that results in the lowest cost given the constraints. The dynamic time warping algorithm (DTW) is a particular application of dynamic programming which tries to optimally align acoustic segments in time, using their respective sequences of feature vectors which represent frames of the original speech (Rabiner and Juang, 1993). There are many possible alignments for any two acoustic segments and the DTW algorithm tries to find the optimal one in an efficient way. The algorithm involves constructing a trellis structure filled with values that indicate the cost of each possible alignment. The next section will describe this process in more detail.

### 4.2.1 Determining the possible alignments

Consider two acoustic segments of speech that have been divided into frames as illustrated in Figure 4.3. Each speech frame is described by a feature vector. In order to find the optimal alignment between the two sequences of frames one should consider all the different alignments possible. These alignments can be represented in a trellis structure as shown in Figure 4.4. The bottom left corner corresponds to the first frame and the top right corner the last frame of both acoustic segments. The first column considers all the frames of

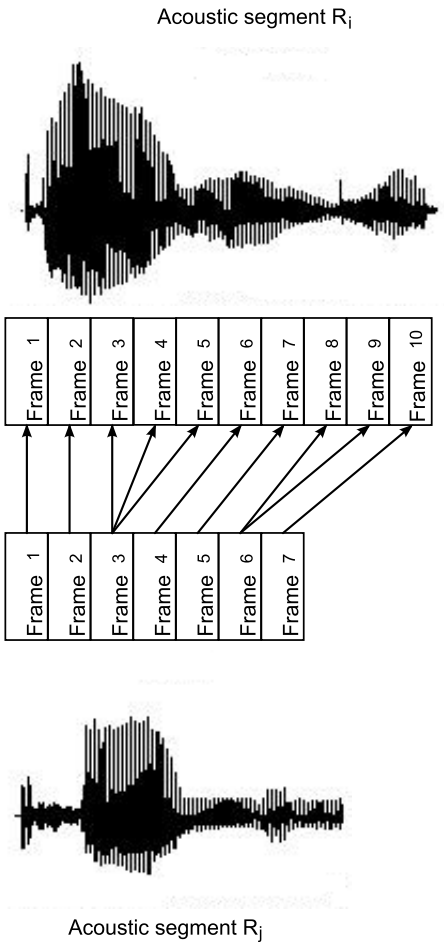



**Figure 4.3:** Example of an alignment between two acoustic segments.

the first acoustic segment that could align with the first frame of the second acoustic segment. The bottom row of the matrix, on the other hand, considers all the frames of the second acoustic segment that may align with the first frame of the first acoustic segment. The best alignment can be represented by a path through the trellis which moves right, up or diagonally. No movement is allowed to the left or down because one of the constraints set by this algorithm is that no alignment back in time is allowed. Furthermore, the path starts at the bottom left and ends at the top right because another constraint is that the start and end frames of the two acoustic segments must be aligned.

The possible alignments are determined iteratively from the bottom left. Except for the first and last row, and first and last column, there are three ways in which the trellis path at each cell can be extended: upwards, diagonally or

horizontally, as shown in Figure 4.4. If the path passes through row $i$ and column $j$, it means that frame $i$ of the first acoustic segment and frame $j$ of the second acoustic segment are aligned. A diagonal path from this point indicates that frame $i + 1$ of the first acoustic segment is aligned with frame $j + 1$ of the second acoustic segment. A vertical move from row $i$ and column $j$ indicates that more than one frame of the first acoustic segment aligns with frame $j$ of the second acoustic segment. Correspondingly, a horizontal move from row $i$ and column $j$ indicates that frame $i$ of the first acoustic segment aligns with more than one frame from the second acoustic segment.

At each iteration all partial paths are extended in all three possible directions. When two paths meet, only the one that accumulates the least cost is retained and the others are discarded. The process completes when all paths have reached the top right cell of the DTW trellis. Only the best scoring path, which has accumulated the lowest overall cost from the bottom left to the top right, will remain and represents the optimal alignment.

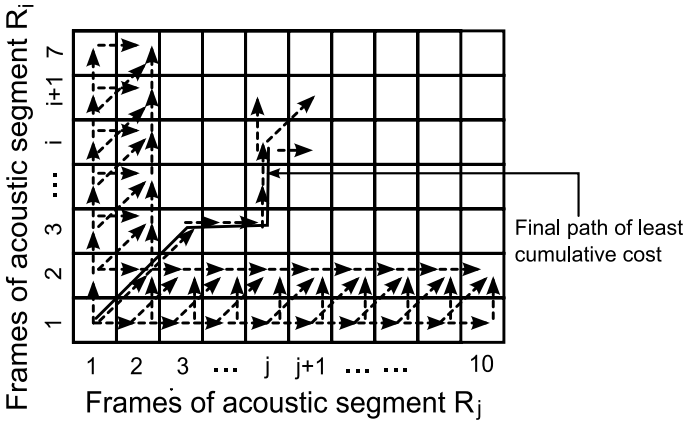



**Figure 4.4:** DTW trellis path example.

A path along the diagonal represents the case where no warping of the time axis occurs. This can only happen when the number of frames of both acoustic segments are the same, and this scenario is very unlikely. One must also keep in mind that even if the best path follows the diagonal, the signals can still be different.

At each point in the trellis, the cumulative similarity of the partial alignment to that point is stored. In the next section the calculation of this similarity and its meaning are explained.

### 4.2.2 Calculating the similarity

To determine the similarity between two feature vectors that represent two frames of speech from two different acoustic segments one uses the Euclidean distance:

$$D_{i,j}^{local} = \sqrt{(v_{i,1} - v_{j,1})^2 + (v_{i,2} - v_{j,2})^2 + ... + (v_{i,N} - v_{j,N})^2}$$

where $v_i$ and $v_j$ are two feature vectors with $N$ dimensions. For the purpose of this thesis, the value will be referred to as the local distance between two feature vectors, while the accumulated distance will be referred to as the global distance and is calculated along a (partial) alignment as follows:

$$D_{i,j}^{global} = D_{i,j}^{local} + \min\left(D_{i-1,j-1}^{global}, D_{i,j-1}^{global}, D_{i-1,j}^{global}\right)$$

where it is assumed that $i > 0$ and $j > 0$. For the first column the calculation differs as follows:

$$D_{i,0}^{global} = D_{i,0}^{local} + D_{i-1,0}^{global}$$

where it is assumed $i > 0$. Similary, for the first row, the global similarity is calculated as follows:

$$D_{0,j}^{global} = D_{0,j}^{local} + D_{0,j-1}^{global}$$

where it is assumed $j > 0$. Finally, to start the DTW recursion, the bottom left entry is calculated as follows:

$$D_{0,0}^{global} = D_{0,0}^{local}$$

Figure 4.5 illustrates this process. The calculation starts at the bottom left corner of the shown trellis. Arrows indicate possible directions in which the calculation can be extended at each iteration. The solid line indicates the best alignment (smallest distance) among the possible options.

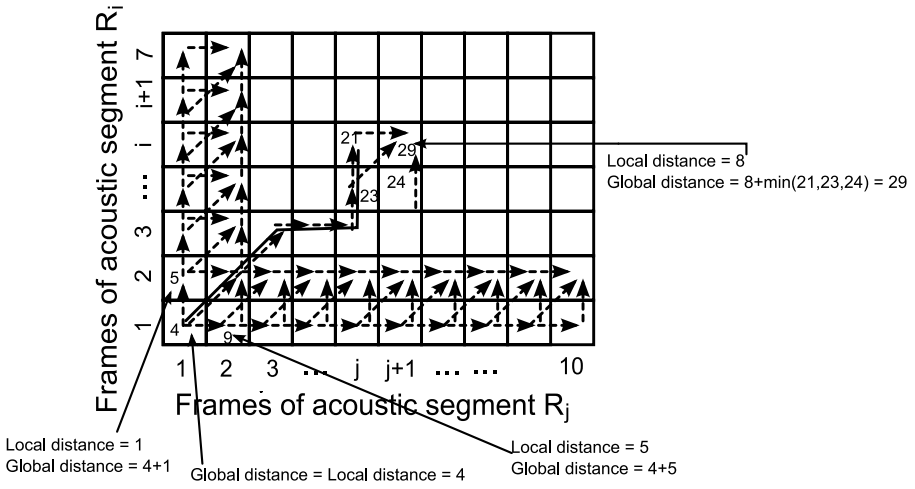



**Figure 4.5:** An illustration of the recursive calculation of the similarity of two acoustic segments, using DTW.

## 4.3 Relationship between the DTW and Viterbi algorithms

There are strong similarities between the DTW algorithm and the Viterbi algorithm. The Viterbi algorithm is used with HMMs and its purpose is to find the optimal state sequence in the HMM, given a sequence of observation vectors. The different paths that can be taken between the states can also be represented with a trellis structure much like that used by the DTW algorithm. However, instead of looking for the path that would yield the lowest cost, as in the case with DTW, the path of the sequence of states that yield the highest probability is found by the Viterbi algorithm. The entries in the trellis structure depend heavily on the configuration and the topology of the HMM. Hence, in the case of the HMM, the Viterbi algorithm determines the optimal alignment between a single sequence and the states of an HMM. The DTW algorithm, on the other hand, determines the alignment between two observation sequences.

## 4.4 Clustering

The DTW algorithm is applied to every possible pair of acoustic segments, resulting in a lower triangular matrix of global similarities, as illustrated in

Figure 4.1. The number of rows and columns of this matrix is equal to the number of acoustic segments determined in the segmentation stage. Initially, the number of acoustic clusters is equal to the number of acoustic segments and each acoustic cluster consists of a single acoustic segment. The rows and columns of the matrix represent these initial acoustic clusters. The first iteration of the clustering process merges the two most similar acoustic segments to form an acoustic cluster that contains two acoustic segments. The matrix can now be updated, with one less row and column, as illustrated in Figure 4.6. The process continues until a predetermined number of acoustic clusters has been reached. Although it is clear how the similarity between two clusters each containing only a single acoustic segment is determined, it is less clear how the similarity between two clusters each containing multiple segments can be determined.

Strategies by which this can be achieved have been considered, and are illustrated in Figure 4.7. These similarities can be used to determine the similarity between two acoustic clusters, as required in subsequent iterations of the clustering algorithm. The most similar pair of acoustic segments from each of the acoustic clusters or the average similarity of all acoustic segments between two acoustic clusters can be used as measures of the overall cluster similarity. The next two sections will explain how these similarities are calculated using the entries in the similarity matrix.

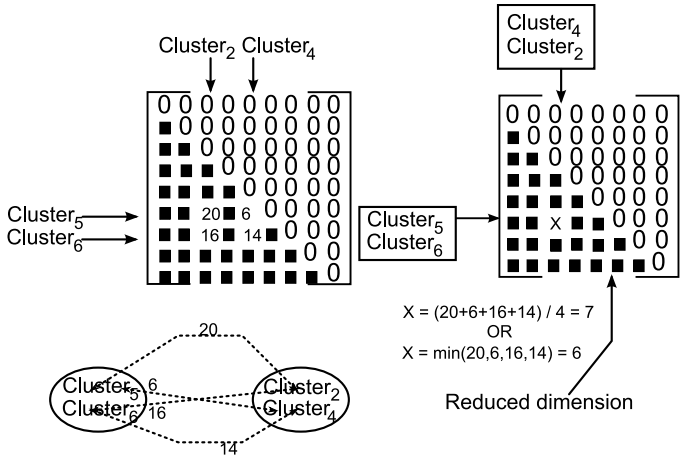



**Figure 4.6:** Illustration of the matrix with distances between acoustic segments/clusters, and the updated matrix after merging clusters five and six, and clusters two and four.

### 4.4.1 Most similar clusters

This approach uses the most similar pair of acoustic segments, one from each cluster to represent the overall similarity between clusters, as illustrated in the top half of Figure 4.7. The similarity is quantified using equation 4.4.1 which is a minmum distance measure between the two clusters.

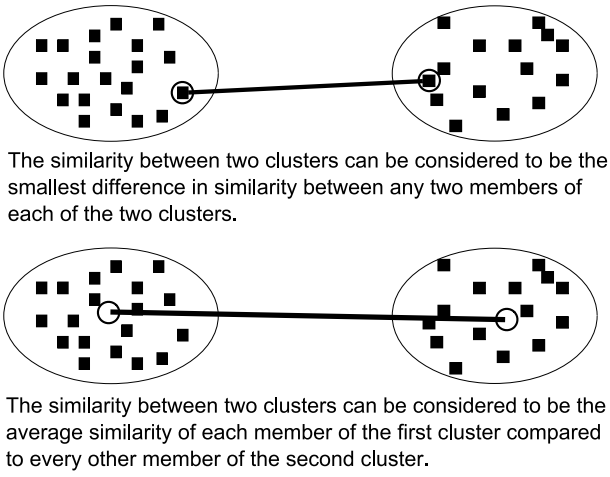



**Figure 4.7:** Strategies for calculating the distance between two clusters each containing multiple acoustic segments.

The similarity is calculated as follows. Let $v_1$, $v_2$, ... $v_N$ be the $N$ members of the first cluster and $u_1$, $u_2$, ... $u_M$ the $M$ members of the second cluster, then

$$Similarity = min(DTW(v_i, u_j)) \text{ for i = 1,2,..N and j = 1,2,..M} \quad (4.4.1)$$

where DTW($v_i$,$u_j$) is the global similarity between two speech segments $v_i$ and $u_j$ as calculated by the DTW algorithm.

### 4.4.2 Average similarity between clusters

In this approach the average similarity between all possible pairs of acoustic segments is used as the final measure of similarity between the two clusters. This is illustrated in the lower half of Figure 4.7. Equation 4.4.2 quantifies the similarity which is an average distance measure between the two clusters.

The similarity is calculated as follows. Let $v_1$, $v_2$, ... $v_N$ be the $N$ members of the first cluster and $u_1$, $u_2$, ... $u_M$ the $M$ members of the second cluster,

then

$$Similarity = \frac{1}{NM} \sum_{i=0}^{N} \sum_{j=0}^{M} DTW(v_i, u_j) \text{ for i = 1,2,..N and j = 1,2,..M} \tag{4.4.2}$$

The average similarity between clusters was chosen as the measure in the experimental evaluation because preliminary experiments indicated that it leads to more evenly sized clusters.

## 4.5 Practical considerations for clustering

### 4.5.1 Speed

The clustering stage takes a considerable amount of time, especially if the segmentation stage produces many segments. One way in which execution speed can be increased is to look for not only the two closest, but maybe the ten closest, acoustic clusters in a single iteration. After each iteration a new lower triangular matrix is generated with the updated similarities between all the acoustic clusters in relation to the newly merged acoustic clusters. The disadvantage of this approach is that two clusters could be merged that would have been merged with different acoustic clusters in the next iteration of the slower process. A reduced dimension matrix is produced after each iteration because acoustic clusters are merged. This means that the number of data kept on disk or in memory becomes less after each iteration. If, for example, one has located the ten closest acoustic clusters in an iteration, then the newly generated square lower triangular matrix will have $10^2$ fewer entries. However, it was found in practice that it was more efficient (in terms of speed) not to reduce the size of the newly generated matrix explicitly, but rather to recalculate the entries in the matrix which are affected by the acoustic cluster merges. The rows and columns represented by the acoustic clusters involved in the merge are marked to prevent their participation in further iterations.

### 4.5.2 Number and size of acoustic segment clusters

The acoustic cluster size is a major hurdle in the clustering stage. Some acoustic clusters were found to grow very large and to “swallow” smaller acoustic clusters, although these appear not to be accoustically very similar. To prevent

acoustic clusters from growing too large, a cap is specified on the number of acoustic segments in an acoustic cluster. If this is reached, then the particular acoustic cluster cannot be part of any further merges. Other acoustic clusters will then have to merge with the second or third best candidates. The disadvantages of placing the cap on the acoustic segment cluster size are that the acoustic clusters were not filled up evenly, and that the cap introduces another parameter into the automatic process.

Finally, the number of clusters resulting from the clustering process must also be considered carefully. If a large number of acoustic clusters is produced, then it could be that acoustically similar acoustic segments are not all clustered together. On the other hand, if a small number of clusters is produced, then it could happen that dissimilar acoustic segments become members of the same cluster. The optimal number of clusters must be determined by experimentation.

## 4.6 Preliminary testing

A small corpus was compiled to test the implementation of the DTW and clustering algorithms. The corpus contained recordings of the author uttering seven repetitions of the following set of eight discrete words: “bat”, “bird”, “budget”, “carrot”, “cat”, “dog”, “party” and “hearty”. These words were chosen to be similar in some cases (e.g. “cat” and “bat”) and dissimilar in other cases. The words were manually labelled as single acoustic segments. In total there are fifty-six sound files. The DTW algorithm was tested using a method called cross-validation. Here one of the seven repetitions of each word is considered a template, and the DTW distances to all the remaining forty-eight words are calculated. A word error rate (WER) was then calculated by determining in how many cases the closest words corresponded to other repetitions of the same word. This was repeated for all combinations of template/test words. These experiments were repeated twice, using different parameterizations for the acoustic feature vectors. First, thirteen MFCC coefficients were augmented by four time domain features, namely energy, average magnitude difference (AMDF), zero-crossing count and autocorrelation. Second, twelve MFCC coefficients were augmented by the frame energy and their first differentials. The results are shown in the last rows of Tables 4.1 and 4.2, respectively.

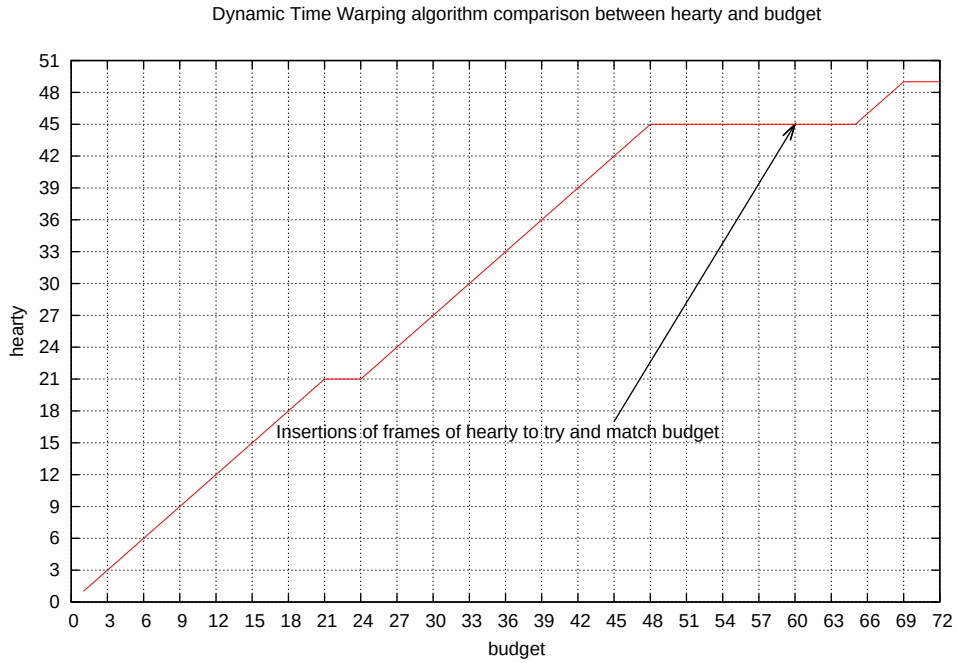



**Figure 4.8:** The DTW algorithm comparison between "hearty" and "budget".

Figure 4.8 shows a DTW alignment between the words "hearty" and "budget". The graph indicates the DTW trellis structure and the line represents the sequence of decisions made to optimally align "hearty" and "budget". This example shows how important the segmentation is. If acoustic segments' boundaries of the same two words aren't labelled correctly, then the acoustic segments will necessarily be misaligned, since one of the constraints is that the first and last frames of the acoustic segments must match.

**Table 4.1:** DTW with 13 MFCCs and four time domain characteristics

| Words | bat | bird | budget | carrot | cat | dog | party | hearty |
|---|---|---|---|---|---|---|---|---|
| bat | 6 | 0 | 0 | 0 | 15 | 0 | 0 | 0 |
| bird | 0 | 21 | 0 | 0 | 0 | 0 | 0 | 0 |
| budget | 0 | 11 | 10 | 0 | 0 | 0 | 0 | 0 |
| carrot | 0 | 0 | 0 | 21 | 0 | 0 | 0 | 0 |
| cat | 0 | 0 | 0 | 0 | 21 | 0 | 0 | 0 |
| dog | 0 | 0 | 0 | 1 | 0 | 20 | 0 | 0 |
| party | 0 | 0 | 0 | 0 | 0 | 0 | 20 | 1 |
| hearty | 0 | 0 | 0 | 0 | 0 | 0 | 10 | 11 |
| Word error rate (WER) = 77.38% | | | | | | | | |

**Table 4.2:** DTW with 12 MFCCs, log energy, MFCC deltas, and log delta

| Phrases | bat | bird | budget | carrot | cat | dog | party | hearty |
|---|---|---|---|---|---|---|---|---|
| bat | 11 | 0 | 0 | 1 | 9 | 0 | 0 | 0 |
| bird | 0 | 21 | 0 | 0 | 0 | 0 | 0 | 0 |
| budget | 0 | 5 | 16 | 0 | 0 | 0 | 0 | 0 |
| carrot | 0 | 1 | 0 | 20 | 0 | 0 | 0 | 0 |
| cat | 0 | 0 | 0 | 0 | 21 | 0 | 0 | 0 |
| dog | 0 | 0 | 0 | 1 | 0 | 20 | 0 | 0 |
| party | 0 | 0 | 0 | 0 | 0 | 0 | 20 | 1 |
| hearty | 0 | 0 | 0 | 1 | 0 | 0 | 16 | 14 |
| Word error rate (WER) = 85.12% | | | | | | | | |

Finally, a classification experiment was carried out, in which each repetition of the eight words was used as a template in turn, and compared with the remaining words. When duplicate configurations are disregarded, this leads to a total of twenty-one classifications of each word. The classification results for the two parameterizations are shown in Tables 4.1 and 4.2. From these results one can conclude that the standard parameterization used by many speech recognition systems (12 MFCCs, energy and deltas) leads to better performance. The word error rate of over eighty-five percent also confirms that the DTW implementation is working.

## 4.7 Summary and conclusion

This chapter has dealt with the clustering of the acoustic segments determined by the previous stage of the algorithm, as described in Chapter 3. An agglomerative clustering algorithm, using a distance measure based on dynamic time warping (DTW), was presented and tested on a small database. This allows the acoustic similarity of two segments of arbitrary lengths to be estimated. The agglomorative clustering algorithm used the average distance between pairs of cluster members as a similarity measure. Clustering continued until a preset number of clusters were attained. The next chapter will describe the use of the resulting clusters to produce a pronunciation dictionary for the provided orthography.

# Chapter 5

# Pronunciation dictionary generation

## 5.1 Overview

The focus of this chapter is the generation of a pronunciation dictionary from the orthographic transcription and the automatically generated acoustic clusters. The automatically generated acoustic clusters are in the form of a list of the acoustic segment boundaries with their acoustic cluster labels, as shown in Figure 5.1. The dictionary will consist of a mapping between words and the sequences of acoustic cluster labels which represent their pronunciation. The example given in Section 3.3.4, can be used to illustrate this. An extract of a typical input file to the dictionary generation stage, based on the example of Section 3.3.4, is illustrated in Figure 5.1. The "G" prefix indicates an acoustic cluster label.

The proposed approach is to proceed by first finding an initial alignment between the acoustic cluster labels and the words of the orthographic transcription. When all training files have been aligned, a mapping is constructed between words and their respective sequences of acoustic cluster labels. This mapping is considered the initial dictionary, and is subsequently refined by iterative re-alignment. Finally, acoustic models are trained and used to further refine the dictionary by means of repeated Viterbi forced alignment. Figure 5.2 presents a flow chart of this process, and each step will be described in more detail in the following sections.

```
#!MLF#
<some_filename>
500000  4100000 G0
4100000  5200000 G1
5200000  6200000 G2
6200000  7000000 G3
7000000  7800000 G4
7800000  9000000 G5
9000000 11100000 G6
11100000 15800000 G7
15800000 16700000 G27
16700000 19500000 G32
19500000 19800000 G29
19800000 20900000 G12
20900000 22100000 G13
22100000 24800000 G14
.
<some_filename>
```

**Figure 5.1:** Extract of an example input file to the dictionary generation stage.

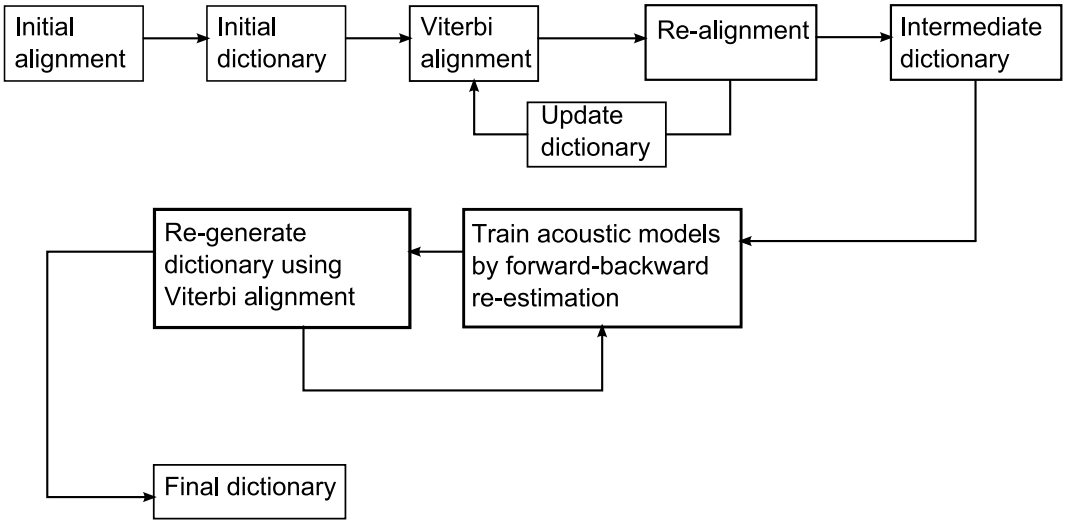



**Figure 5.2:** Flow chart of the dictionary generation stage.

## 5.2 Initial pronunciation alignment

An initial alignment, which is a crude guess as to how the acoustic clusters align with the words in the orthographic transcription, is needed to begin the process. The initial alignment is generated by counting the number of acoustic clusters and the number of words in each training utterance. Then the acoustic clusters are aligned with the words while trying to keep the number of acoustic clusters per word approximately constant. Figure 5.3 presents a visual representation of a hypothetical initial alignment, while the process is presented in more detail in Appendix A. At this point it should be mentioned that there are alternatives to this naïve initialisation approach, such as trying to take the lengths of the words into account. However time did not permit these to be explored.

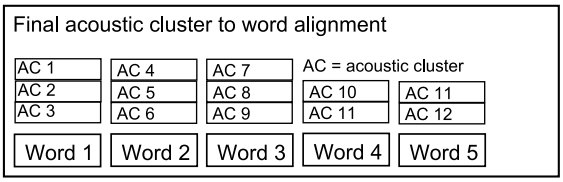



**Figure 5.3:** Example of an initial alignment of acoustic clusters and words.

The procedure illustrated in Appendix A provides an initial mapping between acoustic clusters and words. Now the mapping needs to be refined. The refinement entails changing the alignment between the acoustic clusters and the words in an iterative way, and this is described in the next section.

## 5.3 Refinement of initial pronunciation alignments

Since each word is mapped to a sequence of acoustic clusters, a string of words maps to a string of acoustic cluster sequences. The alignment problem can therefore be reduced to determining whether the current acoustic cluster belongs to the word before or after it in time. Since the sequence of both words and acoustic clusters occurs in a fixed order, an HMM can be used as an appropriate statistical model with which to perform this alignment. The states

of each HMM correspond to the words and the sequence of acoustic clusters to the observations of the HMM. Each utterance can then be represented by an HMM comprising of a sequence of word HMMs. Finding the optimal alignment is reduced to finding the state sequence that maximises the probability of the observation sequence. This can be achieved by means of the Viterbi algorithm.

### 5.3.1 HMM alignment using the Viterbi algorithm

In order to model the sequential nature of the orthography, a left-to-right HMM structure with no skips was chosen, as shown in Figure 5.4. In terms of the Viterbi trellis, also illustrated in Figure 5.4, this means that the alignment starts at the top left corner, and proceeds to the bottom right, with no movement to the left permitted. For each successive observation (acoustic cluster), the path through the HMM will either stay in the current HMM state or move on, but never jump backwards. This condition is enforced by the configuration of the HMM, and indicated by the arrows in Figure 5.4.

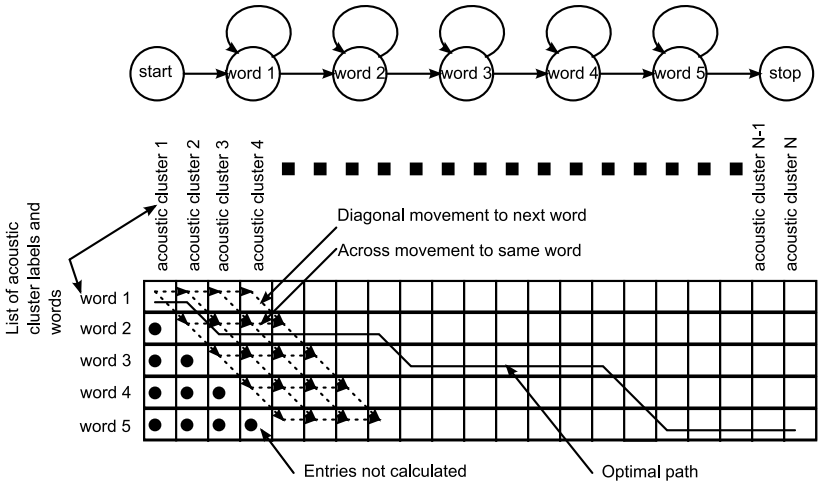



**Figure 5.4:** Example HMM configuration and corresponding trellis structure.

In the trellis structure shown in Figure 5.4, the left-to-right nature of the HMMs means that only movement across or diagonally is allowed. A movement across implies assigning the next acoustic cluster to the same word, while a movement diagonally implies assigning the next acoustic cluster to the next word. Each node of each HMM has associated with it a probability distribution

describing how likely it is for the state to be associated with each acoustic cluster. These are the observation probabilities, and their calculation is considered in more detail in the next section.

### 5.3.2 HMM observation probability distributions

Since the orthography is fixed, one should consider the best alignment to be the one which maximises the conditional probability $P(Cluster_{seq}|Word_{seq})$. The probability is estimated from a relative frequency that is obtained from the most recent alignment. In particular, the probability that cluster $c_i$ is associated with word $w_j$ is calculated as:

$$P(c_i|w_j) = \frac{\text{Number of times } c_i \text{ is aligned with } w_j}{\text{Total number of clusters aligned with } w_j}$$

The totals on the right hand side of the above equation are obtained by accumulating the counts obtained from the most recent alignment of the entire training set with the corresponding acoustic cluster sequences. As an example, the following table shows the number of times each cluster is aligned with each word for a hypothetical example.

**Table 5.1:** Histogram matrix example with hypothetical values.

| Regions | Cluster 1 | Cluster 2 | Cluster 3 | Cluster 4 | Cluster 5 | ... | Total |
|---|---|---|---|---|---|---|---|
| Word 1 | 31 | 49 | 12 | 45 | 101 | ... | $N_1$ |
| Word 2 | 44 | 23 | 77 | 48 | 66 | ... | $N_2$ |
| Word 3 | 11 | 31 | 67 | 79 | 76 | ... | $N_3$ |
| Word 4 | 22 | 51 | 69 | 80 | 32 | ... | $N_4$ |
| Word 5 | 33 | 52 | 71 | 89 | 13 | ... | $N_5$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋱ | |
| Total | $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ | | |

The example in Table 5.1 shows the probability, $P(Cluster_1|Word_1)$, can be estimated as $31/N_1$. For each utterance, a left-to-right HMM in which the states correspond to the words of the utterance and the observations the clusters, was obtained. Uniform transistion probabilities were assumed. Using these HMMs, a Viterbi alignment was performed, which resulted in a new alignment between the word and cluster sequences. The process is described in more detail in the following section. These probabilities are used to perform a Viterbi re-alignment.

### 5.3.3 Viterbi alignment

The Viterbi algorithm is implemented as a two-dimensional trellis structure where the rows represent consecutive words and the columns consecutive acoustic clusters. The elements of the trellis contain the cumulative probabilities of the best alignment to the corresponding word-cluster pair. The first entry in the trellis is given by:

$$ViterbiTrellis[0,0] \;=\; log(P(Cluster_0|Word_0))$$

while the values populating the first row are calculated using the recursion.

$$ViterbiTrellis[0,j] \;=\; log(P(Cluster_j|Word_0)) + ViterbiTrellis[0,j-1]$$

The remainder of the entries in the trellis, where $j > i$, are calculated as follows:

$$\begin{aligned} ViterbiTrellis[i,j] \;&=\; log(P(Cluster_j|Word_i)) \\ &+\; max(ViterbiTrellis[i,j-1], ViterbiTrellis[i-1,j-1]) \end{aligned}$$

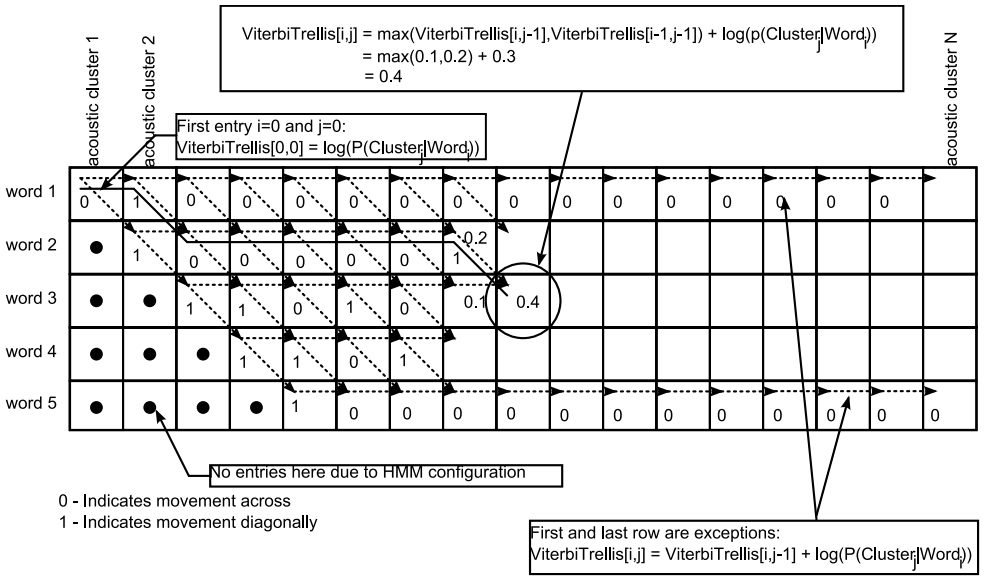



**Figure 5.5:** Calculations performed during Viterbi alignment.

The process is illustrated in Figure 5.5. A zero indicates that the acoustic cluster is assigned to the current word and the path through the HMM stays in

the current state. A one indicates that the acoustic cluster must be assigned to the next word and the path through the HMM moves to the next state. Once the optimal path has been found, the new alignment is used to update the output probability distributions, after which the alignment is repeated. The current iteration's Viterbi alignment is compared with the previous iteration and if there has been any change, then another iteration is performed. In the practical implementation two trellises are kept in memory, one with probabilities as cell entries and the other filled with zeros or ones. Since it is possible that the process could alternate between two or more Viterbi alignments [1], and hence not converge, a cap is placed on the number of iterations.

Now one has an improved alignment between acoustic clusters and words and an optimised associated dictionary. This dictionary is referred to as the intermediate dictionary. The intermediate dictionary will contain many pronunciation variants for each word. In order to remove unlikely variants from the dictionary, a final refinement is performed. In this procedure, acoustic models are trained for the acoustic clusters with the acoustic training data, and pronunciations are pruned from the dictionary by iterative Viterbi alignment.

## 5.4 Final dictionary generation

When one has the intermediate dictionary and the associated alignments, one will generally have many different pronunciations for the same word. This means that the same word is mapped to several different acoustic cluster sequences. It could be that some of these acoustic cluster sequences are rarely associated with a word. So, in this last step of the generation of the pronunciation dictionary one would like to prune out mappings that are not often used. The result of this would be a pronunciation dictionary which is more concise. Figure 5.6 illustrates the process in block diagram form.

### 5.4.1 Training the set of HMMs

In order to obtain acoustic models that can be used for speech recognition, a set of HMMs must be trained. The procedure used to achieve this is illustrated

[1] This was observed in practice.

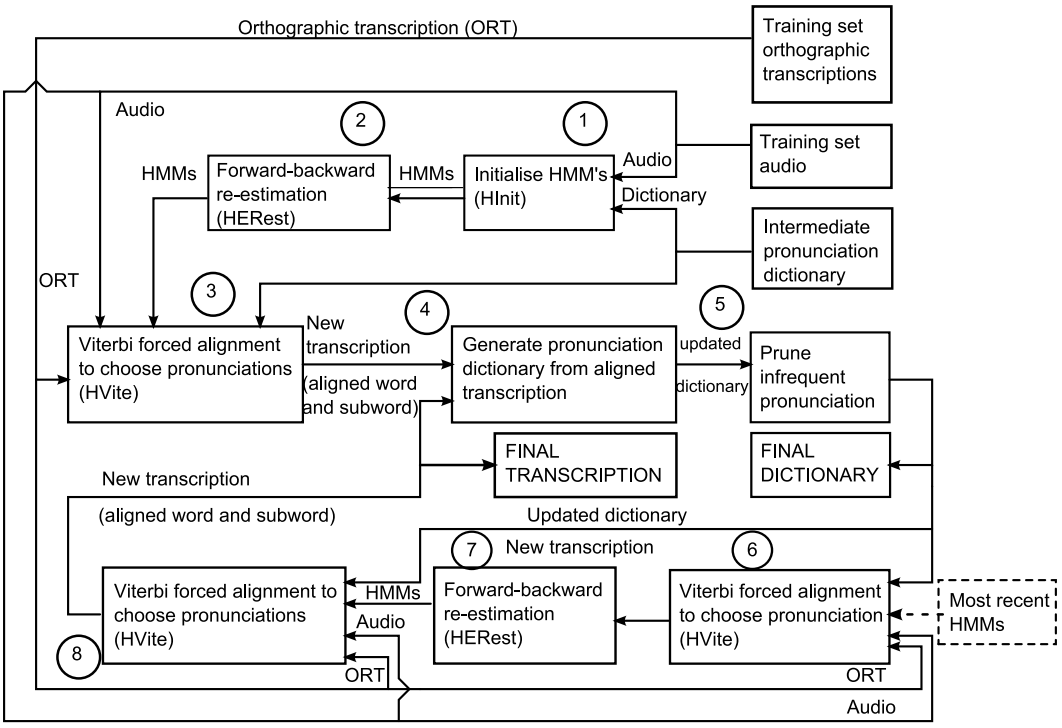



**Figure 5.6:** Procedure followed for final dictionary refinement. HTK tools are indicated in brackets.

in Figure 5.6.

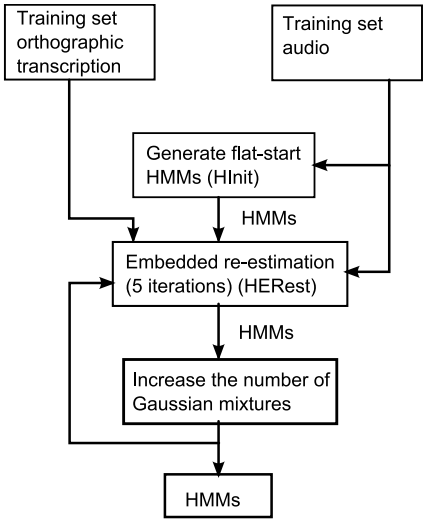



**Figure 5.7:** Training the set of HMMs.

First the intermediate dictionary is used to create initial HMM models using the HTK tools HInit and HERest, as illustrated by steps one and two in Figure 5.6. Single-mixture flat-start context-independent HMMs are initialised from the global mean and variance of the training set audio for each acoustic cluster using HInit. HInit allows initial parameter estimates for HMMs to be obtained from the training data. These initial models are then updated by performing five iterations of embedded re-estimation using HERest. HERest refines the parameter estimates iteratively by performing embedded Baum-Welch re-estimation using the training data. In each case each HMM consists of three states arranged in a left-to-right topology, with a single Gaussian mixture per state. Acoustic observations are parameterized as MFCCs, with appended energy, first and second differentials. The result is a thirty-nine dimensional feature vector.

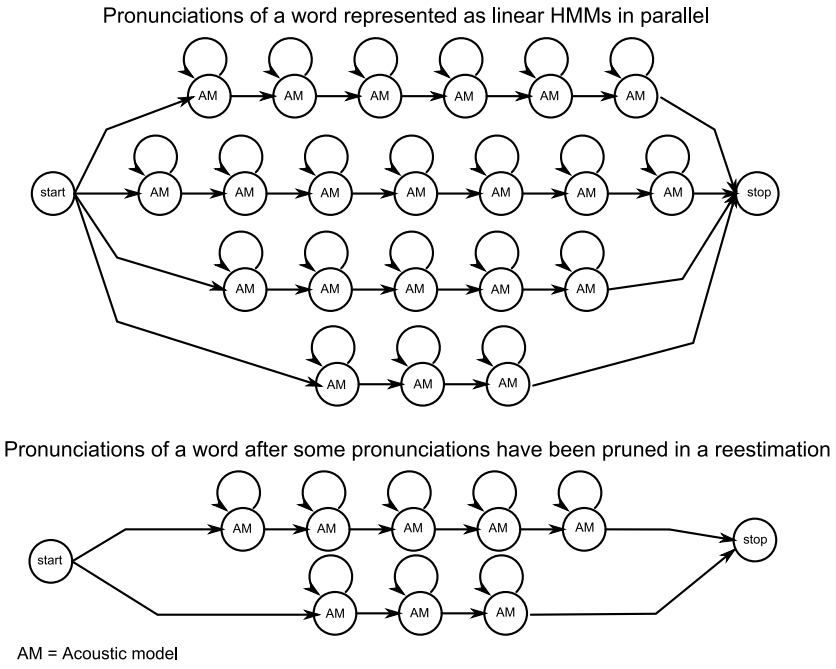



**Figure 5.8:** Pronunciation of a word represented as parallel linear HMMs, as used during forced alignment.

The HMM models obtained in the above process, together with the intermediate dictionary, are then used to perform a forced alignment between the orthography and the acoustic data, using all the pronunciations variants

contained in the dictionary (step three in Figure 5.6). In this process, pronunciation variants are presented in parallel to the Viterbi decoder, as shown in the the top half of Figure 5.8. Not all pronunciation variants will be favoured by the forced alignment, and those rarely used are subsequently pruned from the dictionary in step five of Figure 5.6. In the next section this pruning algorithm will be explained in more detail. The most recently updated HMMs, as well as the new dictionary, are then used to perform further forced alignments, followed by updates to the HMMs as shown in steps six, seven and eight of Figure 5.6. This process repeats until the dictionary no longer changes between successive iterations. Finally, the number of Gaussian mixtures per HMM state is increased from one to two and then four, six and eight where each increase is followed by a further set of five Baum-Welch re-estimation iterations.

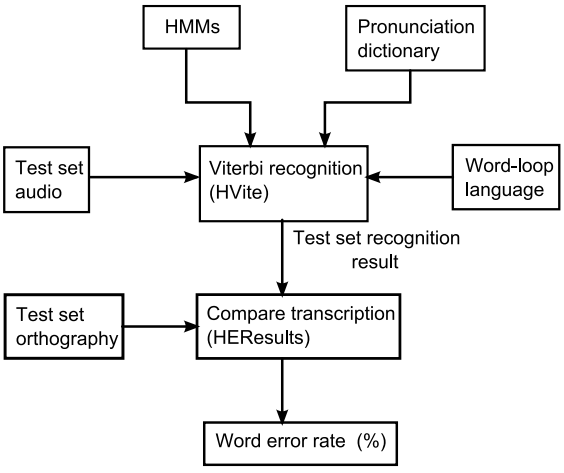



**Figure 5.9:** Procedure used to calculate the word error rate achieved by a trained set of HMMs and associated pronunciation dictionary.

### 5.4.2 Dictionary pruning algorithm

The pruning algorithm plays an important part in the generation of the final dictionary. Due to time constraints, a simple algorithm was devised. The pruning algorithm is governed by a parameter called the pruning threshold. This parameter is not the same as the segmentation threshold referred to in Chapter 2. The optimal value of this pruning threshold can only be determined experimentally. The proposed system is therefore vulnerable to two threshold

values. In general, when a new segmentation threshold is selected, a new pruning threshold has to be determined experimentally. Therefore, the second parameter introduced into the system threatens the notion of completely unsupervised clustering.

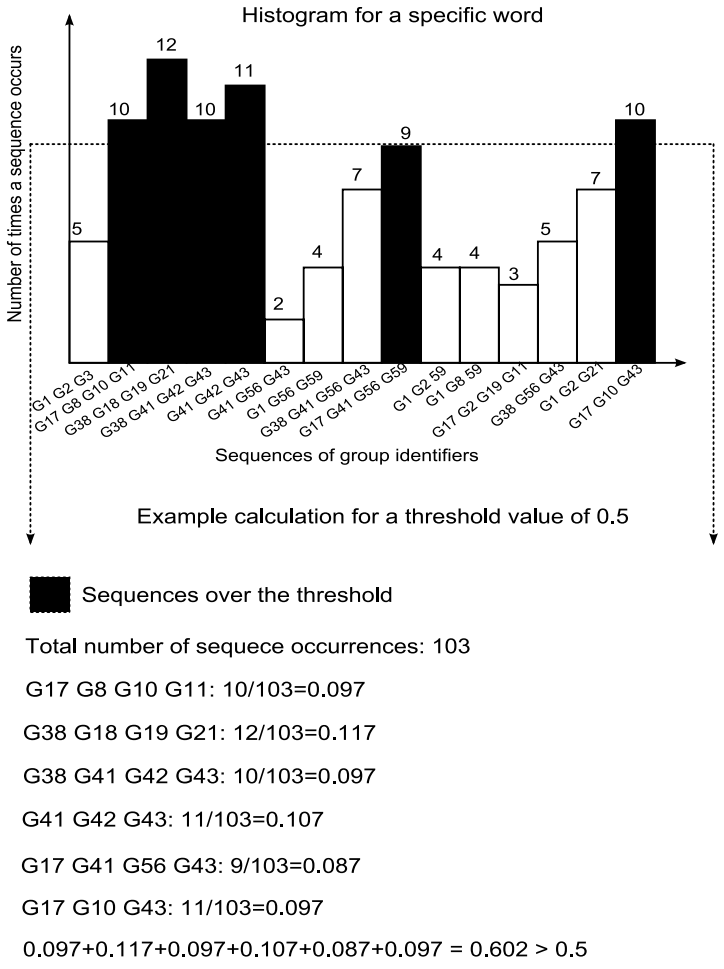



**Figure 5.10:** Example illustrating the dictionary pruning algorithm for a threshold value of 0.5.

The pruning algorithm is illustrated in Figure 5.10 and denoted by step five in Figure 5.6. First, all the unique sequences of acoustic clusters associated with each particular word are tallied (step four in Figure 5.6). The result of this tallying is a histogram which indicates the probability of each sequence of clusters representing the word, i.e. $P(Cluster_{seq}|Word)$. Then, starting with the highest probability sequence, the sequence probabilities are accumulated until this accumulated probability exceeds the pruning threshold.

All pronunciations that form part of this accumulated total are retained, while the remainder are pruned from the dictionary. Hence a pruning threshold of one would keep all pronunciations, while pruning becomes ever more severe as the threshold approaches zero. However, at least one pronunciation is retained for each word. The result is that the word to cluster sequence mappings that occur often in the training set alignment are retained, while infrequent ones are not.

The final dictionary produced by this process can be used to train a new set of acoustic models that can be used in a speech recognition system. The final HMMs are used to perform speech recognition using the procedure shown in Figure 5.9. The final pronunciation dictionary (see Figure 5.6), the final HMMs (see Figure 5.7) and a word-loop grammar, in which any word can follow any other word with equal probability, are used by the HTK tool HVite to perform Viterbi decoding of the test set. The resulting transcription is compared with the test set reference transcription in order to calculate the word error rate (WER) of the system.

## 5.5 Preliminary testing

The dictionary generation stage was tested by simulating "ideal" input from the clustering stage. The ideal data were considered to be the TIMIT SA sentences (see Chapter 2) together with their phonetic transcriptions. Each sentence is repeated by every speaker, which represents an optimistic scenario for the dictionary generation stage because each word is repeated many times. Since the phonetic transcriptions were used, the segmentation stage and clustering stage were skipped. Hence the number of clusters is equal to the number of predefined phonemes in TIMIT. The dictionary generation stage was therefore fooled into thinking that the audio files had been passed through the segmentation and clustering stage by mapping phoneme identifiers to cluster identifiers. This mapping is used to generate an input file for the dictionary generation stage. The process is dubbed phonetic segmentation and clustering.

First the effect of the dictionary pruning threshold on the average number of pronunciations per word in the dictionary was investigated. Figure 5.11 shows that, as expected, the number of pronunciations per dictionary word increases monotonically from a minimum of one as the threshold is varied between zero

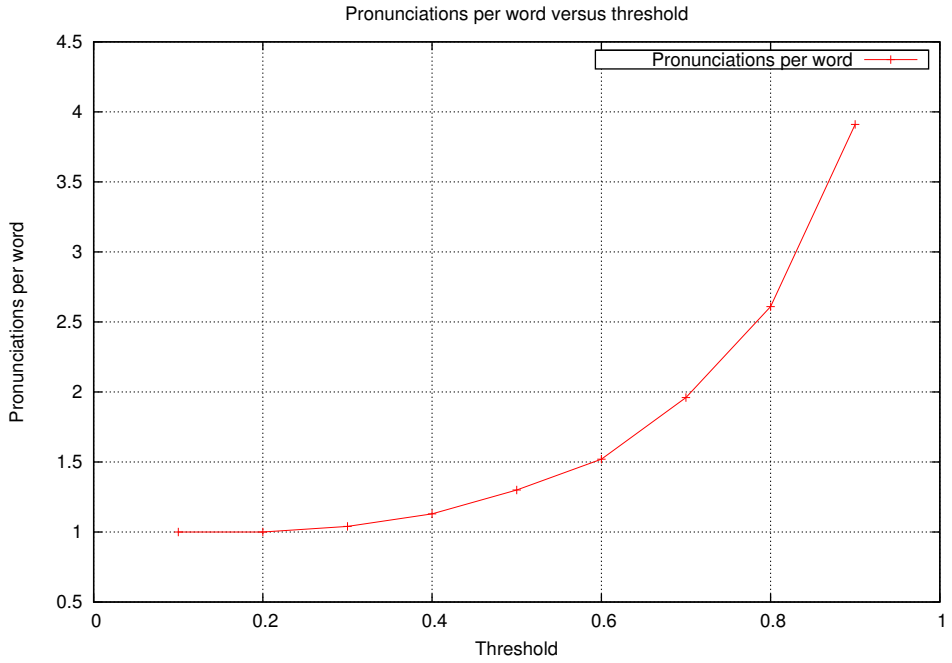



**Figure 5.11:** Dictionary pruning threshold versus pronunciations per word (SA data set and phonetic clusters).

and one. The graph also shows that, without pruning, the TIMIT phonetic transcripts lead to a dictionary with approximately four pronunciations per word.

Next, the effect of dictionary pruning on speech recognition accuracy was investigated. In order to do this, acoustic models are trained as described in Section 5.4.1, using the dictionaries shown in Figure 5.6 as pronunciation dictionaries.

**Table 5.2:** Recognition accuracy using dictionaries obtained for the different pruning thresholds when using phonetic clusters.

| | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Mixtures | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 1 | 90.21 | 90.10 | 90.07 | 90.05 | 90.07 | 89.67 | 89.07 | 85.19 | 84.60 |
| 2 | 92.29 | 92.29 | 92.45 | 92.43 | 92.83 | 92.86 | 93.33 | 91.64 | 90.45 |
| 4 | 93.62 | 93.62 | 93.45 | 93.71 | 93.67 | 93.86 | 94.24 | 93.43 | 93.17 |
| 6 | 94.21 | 94.21 | 94.07 | 94.12 | 94.48 | 94.50 | 94.62 | 93.76 | 93.55 |
| 8 | 94.40 | 94.40 | 94.10 | 94.33 | 94.48 | 94.60 | 94.64 | 94.00 | 93.64 |

The results in Table 5.2 show that the recognition accuracy is always high,

but generally accuracy decreases as the number of pronunciations per word increases with an increase in the pruning threshold. Table 5.2 also illustrates that for one and two mixture systems a more dramatic drop in recognition accuracy was observed compared to that of the higher mixture systems. The pruning threshold made less of a difference in the higher mixture systems. The results of Table 5.2 verify that the dictionary generation and refinement stages are working correctly and as expected. The next chapter investigates the performance of a speech recognition system trained using a dictionary obtained in this way when automatic clustering and segmentation are included.

## 5.6 Summary and conclusion

This chapter has described a procedure which obtains a pronunciation dictionary from the orthographic transcriptions of the audio training set, and the accompanying aligned transcriptions in terms of acoustic clusters. The procedure encompasses the iterative re-alignment, first between these two sequences, and subsequently between the audio data and the acoustic clusters. This procedure, together with the preceeding segmentation and clustering stages, will be evaluated experimentally in the following chapter.

# Chapter 6

# Final evaluation

This chapter will present an experimental evaluation of the techniques described in chapters 3, 4 and 5, by applying them to train speech recognition systems using automatically determined subword units from the TIMIT corpus.

## 6.1 Experimental evaluation using the SA data set

The first set of experiments was performed using the SA sentences for both training and testing, as described in Chapter 2. The purpose of these experiments was to determine how the proposed system would perform when presented with input data that is highly repetitive. The highly repetitive orthography of the SA sentences will provide ample training data for each word and hence represents a very optimistic scenario for the training of HMMs. The first requirement is a well informed decision regarding the segmentation threshold value.

### 6.1.1 Segmentation threshold

The average length, and therefore also the overall number of acoustic segments, can be varied by varying the segmentation threshold, as described in Chapter 3. Figure 6.1 shows how the average segment length changes as a function of the segmentation threshold for the SA training set. As a point of reference, one considers the average length of the phonemes in the phonetic transcriptions,

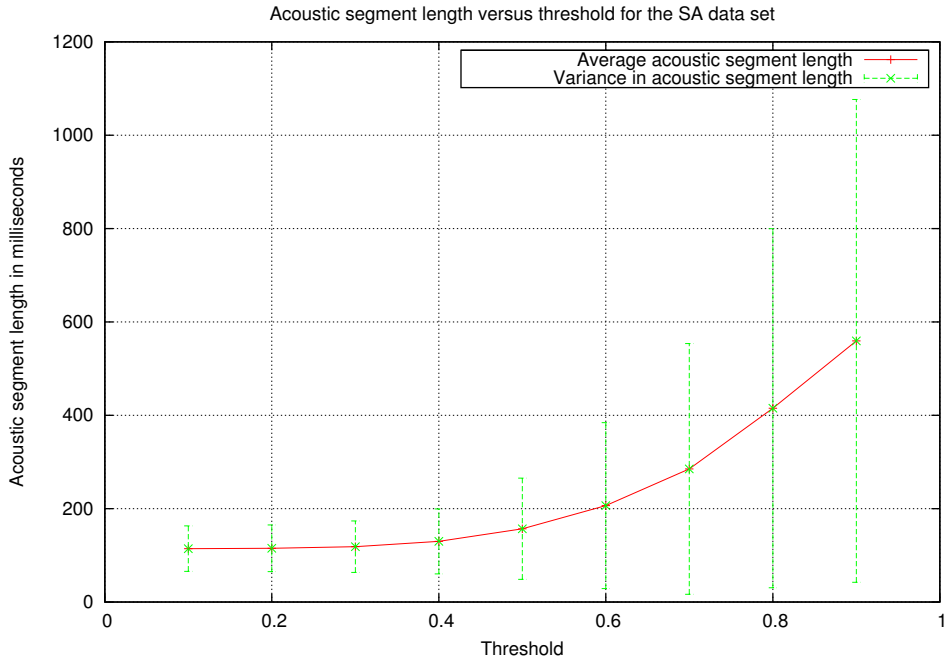



**Figure 6.1:** Acoustic segment length (milli-seconds) versus threshold for the SA data set. Variance is indicated by error bars. The average duration of a phone in the TIMIT transcription is 110ms.

which form part of the TIMIT corpus. This average length is 110ms and therefore one may conclude that a low threshold, in the region of 0.2, would be a good initial value for experimentation.

Figure 6.2 illustrates the total number of segments found in the training set versus the segmentation threshold. A threshold of 0.2 would generate around 8500 acoustic segments.

### 6.1.2 Speech recognition results

Speech recognition results for the SA data set, using a segmentation threshold of 0.2, are presented in Figure 6.3 for eight mixture HMMs. Figure 6.3 indicates the performance of systems trained using automatically determined clusters, as well as phonetic clusters (i.e. the TIMIT phonetic transcriptions). The phonetic clusters serve as a baseline for the automatically determined clusters described in Section 5.5. After automatic segmentation using a threshold of 0.2, the segments were clustered using the approach described in Chapter 4. Systems using 32, 64, 128, 256 and 512 clusters were considered. Next, the

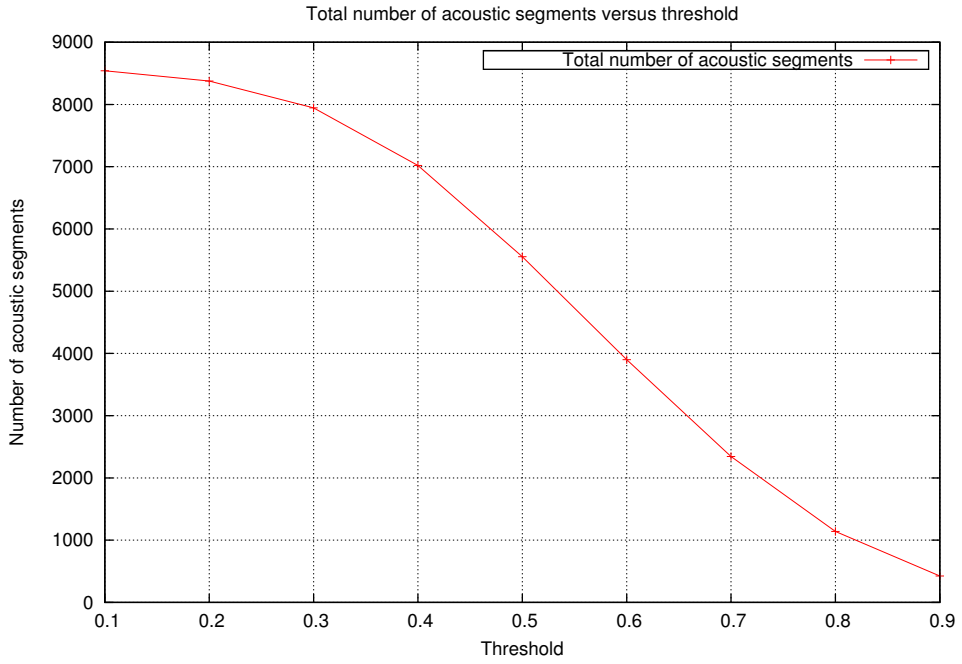



**Figure 6.2:** Total number of acoustic segments versus threshold (SA dataset).

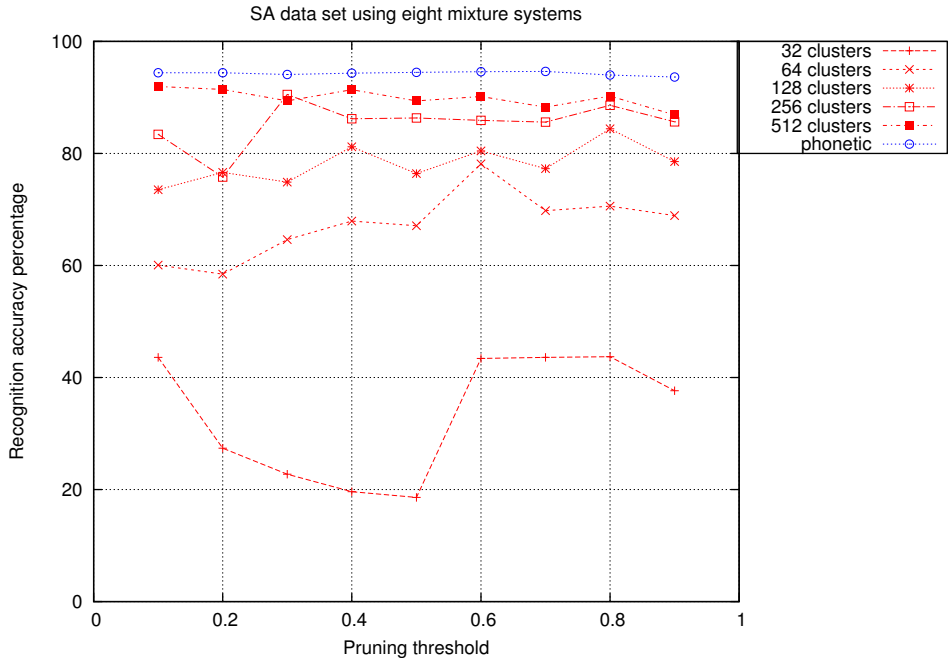



**Figure 6.3:** Results of SA data set using eight mixture systems.

subword transcriptions using these clusters, together with the acoustic training data, were applied to the dictionary generation process described in Chapter 5. The dictionary pruning threshold was varied between a value of 0.1 and 0.9 to produce a variety of systems, whose performance in terms of speech recognition accuracy (word accuracy) are shown in Figure 6.3. Speech recognition was performed using the HTK tool HVite, which performs Viterbi decoding on the audio input using the set of HMMs and the dictionary produced by the process illustrated in Figure 5.6. A word-loop grammar was used as a language model, based on the assumption that the occurrence of each of the 21 words is equally likely. The results for the two, four and six mixture systems are listed in Appendix B and the accompanying graphs are presented in Appendix C. They are similar to those obtained for the eight mixture system. One can see that the phonetic segmentation led to the highest recognition accuracy in almost all cases. This indicates that the pre-defined phonemes in the TIMIT corpus represent the best clusters, and that the automatic segmentation does not perform as well as the phonetic segmentation.

**Table 6.1:** Initial versus final number of subword units for the eight mixture system trained on the SA data set and a dictionary pruning threshold of 0.2.

| Initial versus final number of subword units | |
|---|---|
| After clustering stage | After dictionary generation stage |
| 512 | 32 |
| 256 | 26 |
| 128 | 19 |
| 64 | 14 |
| 32 | 13 |

The results in Figure 6.3 also indicate the effect of varying the number of clusters produced by the clustering stage. Graphs are shown for systems based on 32, 64, 128, 256 and 512 clusters. These numbers refer to the number of clusters produced by the clustering stage. However, during the dictionary estimation stage the pronunciation pruning algorithm discards pronunciations, which generally also leads to a reduction in the number of subword units used by the final system. Table 6.1 indicates how many different subword units remained in the final dictionary for each user-specified number of clusters. The table shows that when the clustering stage produces 512 clusters, only 32 of these remain after dictionary generation. This is comparable to the

number of phonemes defined by TIMIT (48) and leads to the best performing system. It should be kept in mind that the reason why so few clusters remain is probably related to the limited vocabulary of the SA data set.

From the results in Figure 6.3, it is clear that systems with fewer than 256 clusters perform significantly worse than the baseline system using the time phonetic annotations. Secondly, although the baseline was the best performing system, the automatically determined system using 512 clusters achieved performance which was almost as good.

## 6.2 Experimental evaluation using the SI+SX data set

The second set of experiments was performed using the SI+SX data set. This data set is more challenging because the sentences are not repeated among the speakers. This means that building good HMMs for recognition will be more difficult. The same segmentation threshold of 0.2 as used for the SA data set experiment was used again. The results for an eight mixture system are presented in Figure 6.4, while those for two, four and six mixture systems are given in Appendices B and C. As for the SA experiments, a word-loop grammar, in which any of the 2595 vocabulary words may follow each other with equal probability, was used.

### 6.2.1 Speech recognition results

The testing procedure described in Section 6.1 for the SA subset, was repeated for the SI+SX subset of training and test sentences (described in Section 2.3). The SI+SX data set has a larger vocabulary (2595 words) and not all the test words are repeated multiple times in the training data. Furthermore, the specific word sequences are not repeated. This scenario makes it considerably more difficult to train good HMMs for recognition. Even systems trained on the TIMIT phonetic transcriptions have a low recognition accuracy. From the results in Figure 6.4 one can see that the performance of all systems is poor. In general, a recognition accuracy of around 20 percent was achieved. The recognition accuracy curves are also fairly flat compared to those obtained for the SA data set. This is due to two factors: the dictionary pruning threshold and

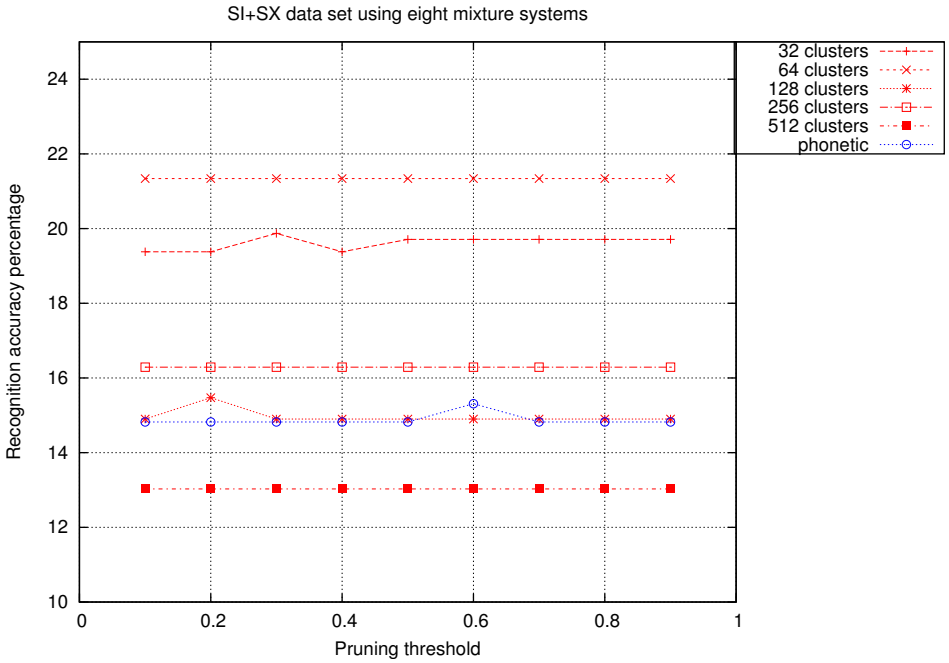



**Figure 6.4:** Results of SI+SX data set using eight mixture systems.

the number of times a specific pronunciation sequence occurs. Both of these influence the dictionary pruning algorithm. Since pronunciations sequences do not often occur, the pruning algorithm has already pruned most pronunciation sequences at a threshold of 0.2.

**Table 6.2:** Initial versus final number of subword units for the eight mixture system trained on the SI+SX data set and a dictionary pruning threshold of 0.2.

Initial versus final number of subword units

| After clustering stage | After dictionary generation stage |
|---|---|
| 512 | 341 |
| 256 | 177 |
| 128 | 93 |
| 64 | 46 |
| 32 | 27 |

Table 6.2 indicates how many different subword units remain in the final dictionary for each user-specified number of clusters. The table shows that, when the clustering stage produces 512 clusters, 341 of these remain after dictionary generation. For 32 user-specified clusters, the dictionary generation stage retains nearly almost all of them. For the case of 64 user-specified

clusters, 46 remained after the dictionary generation stage. This number of clusters is close to the number of defined phonemes in TIMIT and the associated system also achieves the best results overall. For 128 clusters and more, an increasing number are pruned away by the dictionary generation stage.

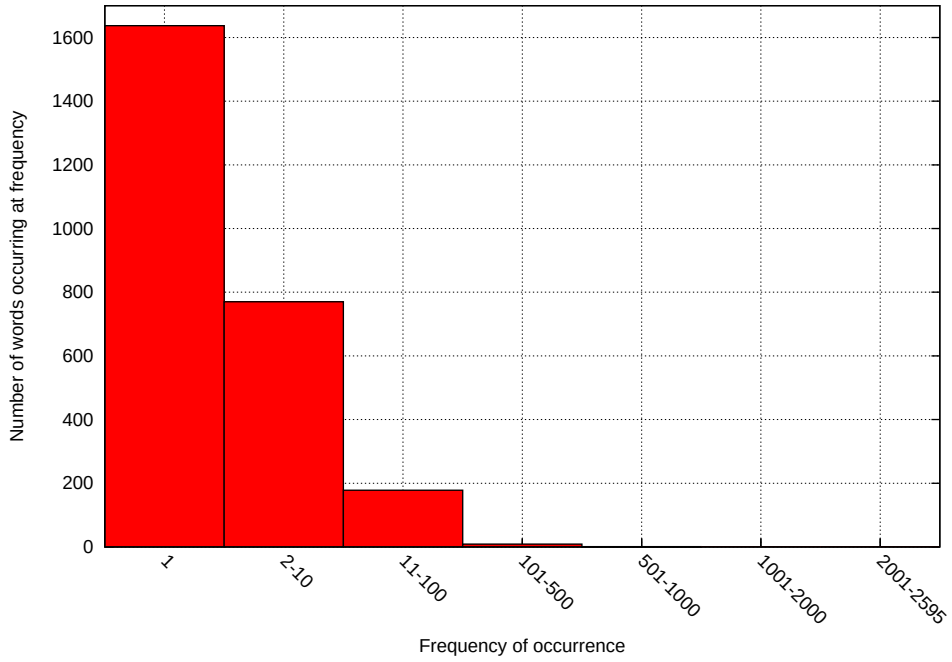



**Figure 6.5:** Histogram indicating the number of words for different frequencies of occurence in the SI+SX training set.

## 6.3 Interpretation of results

For the SA experiments, the performance of the systems improved as the number of clusters increased. Furthermore, it was seen that the overwhelming majority of clusters were pruned during the dictionary generation stage. The same was not true for the SI+SX system, for which best performance was achieved for an intermediate number of clusters (64), and a much smaller proportion of clusters was pruned during dictionary generation.

A major difference between the SA and the SI+SX data sets is the frequency of repetition of the training words in the training sets, as shown in Figure 6.5. The high level of repetition in the SA set allowed many postulated

pronunciations to compete during dictionary generation. This led to a greater degree of dictionary pruning, and to a better performing system. The acoustic models based on the phonetic transcription led to the best results for the SA data set.

For the SI+SX data set, on the other hand, most training words occur only once in the training set. This means there is far less opportunity for competing pronunciations to be generated, and for the subsequent dictionary pruning to remove less well-performing candidates. Nevertheless, acoustic models based on the best configuration (64 clusters) led to better recognition accuracies than models based on the phonetic transcriptions. However, the recognition results were very low overall.

## 6.4 Summary and conclusion

This section has presented experimental results obtained when applying the methods identified and proposed in Chapters 3, 4 and 5 of this thesis to two subsets of the TIMIT database. For the SA subset, it was shown that a system based on phonetic clusters outperforms all systems based on automatically determined clusters, but that the difference in performance was not large. For the SI+SX subset, it was shown that a system based on automatically determined clusters outperformed a system based on phonetic clusters, but that the performance was poor in all cases. The following chapter will provide a more in-depth discussion of these findings.

# Chapter 7

# Summary and conclusions

The goal of this thesis was to develop a method that can automatically generate subword units and an associated dictionary given only the audio data and associated orthographic transcription. The process was divided into stages, in order to allow each stage to be tested and analysed separately. The quality of the dictionary depends largely on the quality of the automatically determined subword units, and on how well the dictionary maps words to subword sequences. Therefore, the success of the proposed solution can be analysed in two parts: the segmentation generation section and the dictionary generation section.

## 7.1 Segmentation

The goal of the segmentation stage is to automatically divide unlabelled speech audio into short, phoneme-like segments that can subsequently be used to determine a pronunciation dictionary. The method proposed by ten Bosch and Cranen (2007) was selected for this purpose. This method identifies boundaries in unlabelled audio by detecting points at which the stream of MFCC vectors change rapidly. Segmentation is followed by clustering, a stage of which the purpose is to group acoustically similar segments together as determined by the segmentation stage. A bottom-up (agglomerative) approach using dynamic time warping (DTW) as a distance measure was selected for this purpose.

The success of the clustering stage, however, depends critically on the quality of the segments themselves. Furthermore, the quality of the segments and resultant clusters critically affect the quality and ultimate success of the dictio-

nary, which uses the clusters of segments as input. By listening to a random selection of the audio segments produced by the segmentation stage, it was concluded that, although sometimes the segments appeared to be plausible subword units, sometimes they were not. Badly-formed segments most certainly have had a detrimental effect on the success of the overall approach. However, time did not permit this issue to be investigated with more rigour. Instead, the improvement of the quality of the segments remains the subject of future work.

## 7.2 Clustering

As already mentioned in the previous section, a bottom-up (agglomorative) clustering algorithm was used to identify groups of similar acoustic segments. The dynamic time warping (DTW) alignment cost between two acoustic segments was used to measure their similarity for purposes of clustering. The clustering algorithm was verified, by means of a specially prepared small speech database, to be working correctly. When the algorithm was applied to the TIMIT data, it was found that some clusters did, indeed, contain segments that sounded similar (this was discovered during informal listening tests), but there were some which seemed to contain a less meaningful collection of sounds. This may be due to the presence of many poor quality segments, i.e. a failure of the preceding segmentation stage. However, the performance of the clustering algorithm itself can be improved by, for example, modifying the distance measure (see Section 8.2). It was noticed, for example, that some clusters had a tendency to grow very large, while some remained very small. Time, unfortunately, did not permit these issues to be explored further.

## 7.3 Dictionary generation

While the segmentation and clustering stages were based on known and published approaches, the procedure for the automatic determination of a pronunciation dictionary is, to the best knowledge of the author, a new approach.

The dictionary is determined by an iterative process of alignment between the automatically determined subword transcription, the associated orthographic transcription, and the corresponding audio data. Subword and or-

thographic transcriptions are aligned by modelling the orthography as a hidden Markov model (HMM), where the subword units are the observations and the states of the HMM are the words. The resulting dictionary is used to align the possible pronunciations with the audio, and thereby discard poorly matching pronunciation variants. The process is iterated until some degree of convergence is achieved.

When this dictionary generation process is presented with the true TIMIT phonetic transcriptions, instead of the automatically determined subword units, a pronunciation dictionary containing reasonable pronunciations (according to informal inspection) was determined. When presented with the automatically determined subword units, the dictionary was difficult to analyse, but did lead to a working system. However, once again further analysis is needed, which unfortunately, time did not permit.

## 7.4 Overall system

The overall approach was evaluated by testing it using two subsets of the TIMIT corpus.

The first, termed the "SA" system, used the two phonetically rich sentences repeated by every speaker as training and testing material. Although there is no speaker overlap between the test and train sets, the same two sentences constituted both. This is a very optimistic testing scenario, since the vocabulary is small (21 words) and each word in the training set is repeated many times (462 times), albeit by different speakers. In this testing scenario it was found that a system trained on automatically determined subword units could achieve a performance nearly as good as one trained on the true TIMIT phonetic transcriptions. Due to the small vocabulary and the high repetition rate of training words, the word accuracies achieved by these systems were rather high.

The second subset of the TIMIT corpus was drawn from the SI and the SX sentences, which are also phonetically rich, but which are repeated by only seven speakers or only once each for the SX and SI sentences respectively. The training and testing subsets were chosen in such a way as to ensure a closed vocabulary, i.e. that each word in the test set also occurs at least once in the training set. Systems trained on this SI+SX subset exhibited much poorer

performance than those based on the SA subsets. However, the best systems based on automatically determined subword units were able to outperform those based on the TIMIT phonetic transcriptions. Notwithstanding the low word accuracies, this is a very promising result.

## 7.5 Overall conclusion

It was possible to obtain working speech recognition systems using only the orthographic transcriptions and the audio data as input. In particular, no pronunciation dictionary or other subword information was employed. The overall system is complex, and time did not permit thorough testing and analysis. However, the limited test results are extremely promising. If the techniques described and proposed in this thesis can be further analysed and refined, it could be an important step for the development of speech recognition systems for under-resourced languages or dialects, for which the extensive phonetic resources conventionally required are not available.

# Chapter 8

# Recommendations

There are many aspects of the work presented in this thesis that warrant further analysis or improvement. Some of these are suggested in the following sections.

## 8.1 Segmentation

The segmentation algorithm of the method proposed by this thesis uses the characteristics of its frequency spectrum. There are other methods that combine techniques of the time domain as well as the frequency domain to generate segments. It may be fruitful to evaluate some of these techniques. Furthermore, a better analysis of the segments produced by these methods would allow insight to be gained into the operation of the overall system.

## 8.2 Clustering

While the use of dynamic time warping (DTW) has, in general, been a successful way to gauge similarity between two acoustic segments, the implementation of this is currently naïve. For example, the method allows extreme warping of the time axis between the two segments. As an improvement, it may be advantageous to limit the degree to which the axes may be warped, either by imposition of hard boundaries in the alignment matrix, or by increasing the cost of non-linear warping of the time axis. See Figure 8.1 for an illustration of this idea.

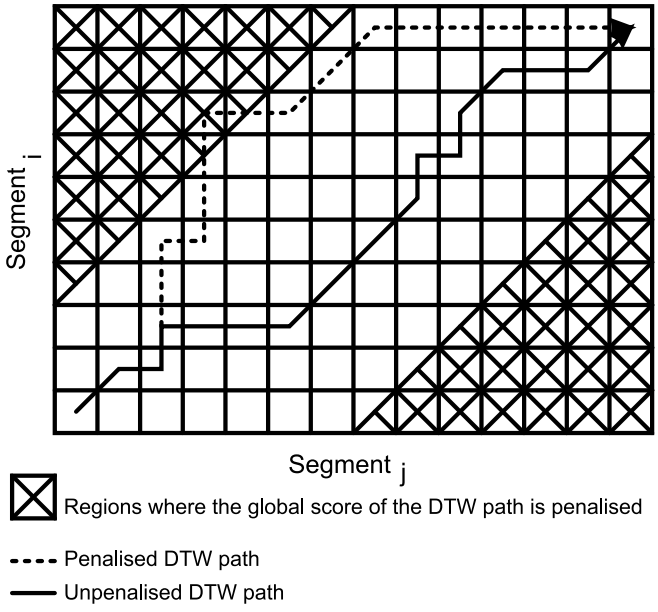



**Figure 8.1:** DTW path penalty recommendation.

Another aspect of the clustering procedure that could be improved is its implementation. Currently, a very large square matrix, whose size is proportional to the square of the number of segments produced by the segmentation stage, is held in memory. Although this approach was practical for the TIMIT data set, it is not practical for larger data sets. It would be better if the clustering process were to be parallelised. The clustering could then be carried out on a multi-core computing system.

## 8.3 Parameter tuning

The overall procedure proposed by this thesis has three main parameters for which suitable values must be provided: the segmentation threshold, the number of clusters and the dictionary pruning threshold. In addition, there are many "hidden parameters", such as the optimal number of HMM re-estimations per iteration during dictionary generation, which have not been explored. A much more extensive set of experiments is required in order to obtain good general values for these parameters. It may also be possible to obtain empirical relationships between known attributes of the desired system (such as the vocabulary size and the number of words in the training set) and some of these

parameters, thereby reducing the risk of using arbitrary values.

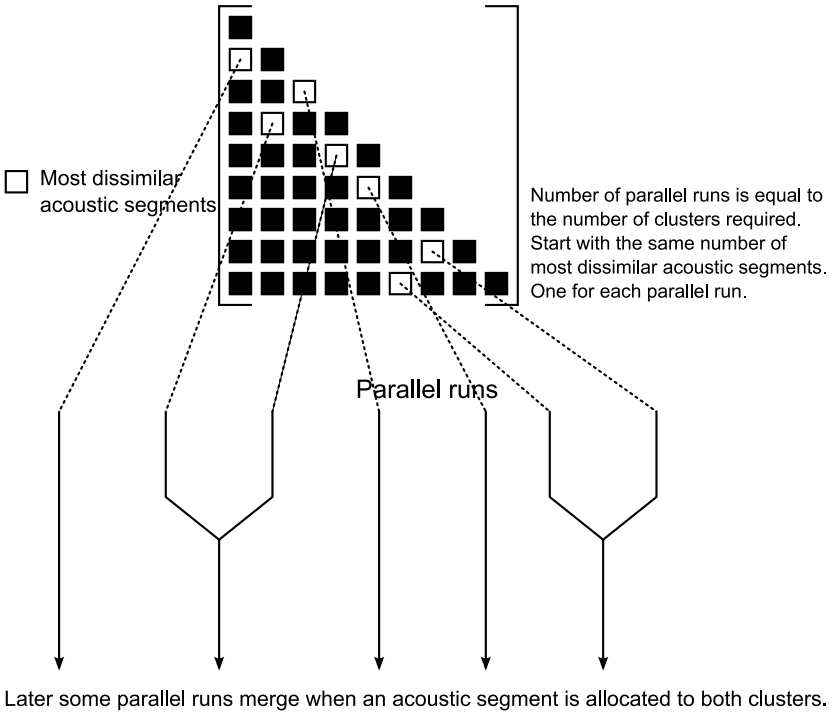



**Figure 8.2:** Parallelisation of the clustering.

## 8.4 Technical improvements

The method can also be improved on in some technical aspects, such as speed and scalability. The clustering algorithm is very memory intensive, because the original matrix is kept in memory. The longest processing time is also spent in this stage. Currently, it takes several days to generate a dictionary for a reasonably sized training set, like TIMIT. Another recommendation would be to speed the algorithm used for the clustering. It may be possible to subdivide the training set and run individual clustering steps in parallel. In order to do this one could begin by finding a number of dissimilar clusters. Then, using these dissimilar clusters as start clusters, each parallel run starts to cluster acoustic units. Subsequently, the results of these clustering processes are merged. Figure 8.2 illustrates this recommendation.

## 8.5 Outer loop

Finally, there is the possibility of adding an outer loop to the proposed procedure. For example, the best final system can be used to resegment the audio into a new set of segments, which are used to derive a new dictionary. Alternatively, an initial system with a relatively small number of clusters can be used to bootstrap systems with an increasing number of clusters. In this way the potentially poor quality initial alignment between the orthographic and subword transcription, which is used to generate the initial dictionary, might be improved.

# Bibliography

Ahmadi, S. and Spanias, A. (1999). Cepstrum-based pitch detection using a new statistical v/uv classification algorithm. *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 3, pp. 333–338. (Cited on page 9.)

Arcienega, M. and Drygajlo, A. (2002). Robust voiced-unvoiced decision associated to continuous pitch tracking in noisy telephone speech. In: *Proceedings of ICSLP*. Denver, USA. (Cited on page 9.)

Aversano, G., Esposito, A. and Marinaro, M. (2001). A new text-independent method for phoneme segmentation. In: *Proceedings of 44th IEEE Midwest Symposium on Circuits and Systems*. Dayton, USA. (Cited on page 10.)

Everitt, S.B. (1993). *Cluster Analysis*. 3rd edn. Cambridge University Press. (Cited on page 19.)

Gajjar, M., Govindarajan, R. and Sreenivas, T. (2008). Online unsupervised pattern discovery in speech using parallelization. In: *Proceedings of Interspeech*. Brisbane, Australia. (Cited on page 10.)

Golipour, L. and O'Shaughnessy, D. (2007). A new approach for phoneme segmentation of speech signals. In: *Proceedings of Interspeech*. Antwerp, Belgium. (Cited on page 10.)

Loukina, A., Kochanski, G., Shih, C., Keane, E. and Watson, I. (2009). Rhythm measures with language-independent segmentation. In: *Proceedings of Interspeech*. Brighton, UK. (Cited on page 9.)

Murthy, H. and Gadde, V. (2003). The modified group delay function and its application to phoneme recognition. In: *Proceedings of ICASSP*. Hong Kong. (Cited on page 10.)

Muscariello, A., Gravier, G. and Bimbot, F. (2009). Audio keyword extraction by unsupervised word discovery. In: *Proceedings of Interspeech*. Brighton, UK. (Cited on page 10.)

Park, A. and Glass, J. (2005). Towards unsupervised pattern discovery in speech. In: *Proceedings of ASRU*. Cancun, Mexico. (Cited on page 10.)

Park, A. and Glass, J. (2008). Unsupervised pattern discovery in speech. *IEEE Transactions on Speech and Audio Processing*, vol. 16, no. 1, pp. 186–197. (Cited on page 10.)

Rabiner, L. and Juang, B.H. (1993). *Fundamentals of Speech Recognition*. 1st edn. Prentice Hall. (Cited on page 20.)

Singh, R., Raj, B. and Stern, R.M. (2002). Automatic generation of subword units for speech recognition systems. *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 2, pp. 89–99. (Cited on pages ix, 1, 2, 3, and 9.)

ten Bosch, L., Baayen, H. and Ernestus, M. (2006). On speech variation and word type differentiation by articulatory feature representations. In: *Proceedings of ICSLP*. Pittsburg, USA. (Cited on page 10.)

ten Bosch, L. and Cranen, B. (2007). A computational model for unsupervised word discovery. In: *Proceedings of Interspeech*. Antwerp, Belgium. (Cited on pages 10, 11, 12, 14, 15, 17, and 53.)

Tolba, H. and O'Shaugnessy, D. (1998). Robust automatic continuous speech recognition based on a voiced-unvoiced decision. In: *Proceedings of ICSLP*. Syndey, Australia. (Cited on page 9.)

Torres-Carrasquillo, P., Reynolds, D. and Deller, J. (2002). Language identification using gaussian mixture model tokenization. In: *Proceedings of ICASSP*. Orlando, Florida. (Cited on page 11.)

Zolnay, A., Schlüter, R. and Ney, H. (2003). Extraction methods of voicing feature for robust speech recognition. In: *Proceedings of Eurospeech*. Geneva, Switzerland. (Cited on page 9.)

# Appendices

# Appendix A

# Initial alignment calculation

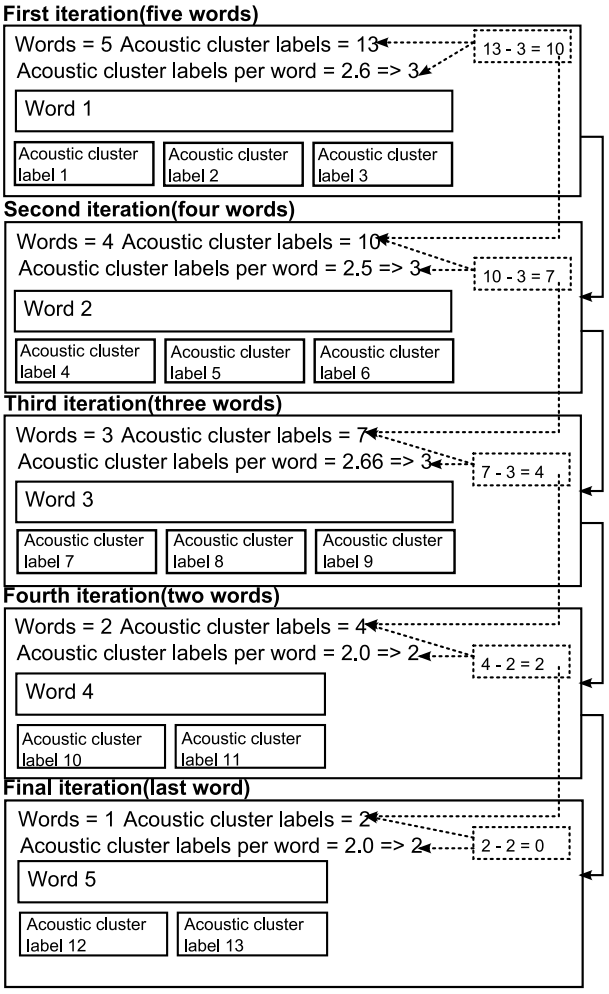



**Figure A.1:** Initial alignment calculation.

# Appendix B

# Automatic segmentation results in table format

**Table B.1:** Eight mixture systems, SA sentences - Recognition accuracy percentage as a function of the specified number of clusters and pruning threshold.

| Number of clusters | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 512 | 91.95 | 91.43 | 89.38 | 91.38 | 89.38 | 90.17 | 88.26 | 90.24 | 86.88 |
| 256 | 83.40 | 75.79 | 90.52 | 86.19 | 86.33 | 85.90 | 85.60 | 88.60 | 85.64 |
| 128 | 73.52 | 76.62 | 74.88 | 81.19 | 76.40 | 80.45 | 77.29 | 84.40 | 78.55 |
| 64 | 60.07 | 58.45 | 64.62 | 67.93 | 67.10 | 78.14 | 69.79 | 70.60 | 68.90 |
| 32 | 43.60 | 27.38 | 22.74 | 19.62 | 18.60 | 43.40 | 43.60 | 43.71 | 37.64 |

**Table B.2:** Eight mixture systems, SI and SX sentences - Recognition accuracy percentage as a function of the specified number of clusters and pruning threshold.

| Number of clusters | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 512 | 13.03 | 13.03 | 13.03 | 13.03 | 13.03 | 13.03 | 13.03 | 13.03 | 13.03 |
| 256 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 |
| 128 | 14.90 | 15.47 | 14.90 | 14.90 | 14.90 | 14.90 | 14.90 | 14.90 | 14.90 |
| 64 | 21.34 | 21.34 | 21.34 | 21.34 | 21.34 | 21.34 | 21.34 | 21.34 | 21.34 |
| 32 | 19.38 | 19.38 | 19.87 | 19.38 | 19.71 | 19.71 | 19.71 | 19.71 | 19.71 |

**Table B.3:** Six mixture systems, SA sentences - Recognition accuracy percentage as a function of the specified number of clusters and pruning threshold.

| Number of clusters | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 512 | 91.00 | 90.64 | 88.83 | 91.62 | 90.64 | 89.00 | 88.45 | 89.07 | 86.52 |
| 256 | 82.88 | 83.05 | 90.40 | 85.98 | 86.33 | 85.31 | 86.05 | 88.24 | 82.98 |
| 128 | 72.88 | 76.36 | 74.43 | 81.69 | 77.07 | 80.43 | 77.38 | 84.74 | 77.67 |
| 64 | 59.62 | 59.50 | 64.21 | 67.33 | 67.74 | 77.81 | 68.93 | 69.19 | 69.26 |
| 32 | 45.38 | 26.93 | 22.55 | 20.07 | 18.12 | 44.07 | 43.38 | 44.60 | 36.40 |

**Table B.4:** Six mixture systems, SI and SX sentences - Recognition accuracy percentage as a function of the specified number of clusters and pruning threshold.

| Number of clusters | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 512 | 13.27 | 14.98 | 13.27 | 13.27 | 13.27 | 13.27 | 13.27 | 13.27 | 13.27 |
| 256 | 16.04 | 16.04 | 16.04 | 16.04 | 16.04 | 16.04 | 16.04 | 16.04 | 16.04 |
| 128 | 14.82 | 14.82 | 14.82 | 15.39 | 14.82 | 14.82 | 14.82 | 14.82 | 14.82 |
| 64 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 |
| 32 | 19.30 | 19.14 | 19.30 | 19.30 | 20.11 | 19.63 | 19.63 | 19.63 | 19.63 |

**Table B.5:** Four mixture systems, SA sentences - Recognition accuracy percentage as a function of the specified number of clusters and pruning threshold.

| Number of clusters | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 512 | 90.43 | 90.40 | 89.24 | 91.86 | 96.62 | 88.02 | 87.62 | 86.69 | 85.24 |
| 256 | 81.71 | 81.55 | 89.17 | 84.90 | 84.17 | 84.55 | 86.98 | 86.83 | 81.24 |
| 128 | 73.83 | 75.71 | 73.95 | 79.69 | 74.02 | 72.93 | 78.43 | 83.62 | 76.12 |
| 64 | 57.48 | 58.29 | 62.29 | 77.24 | 67.57 | 68.79 | 65.83 | 71.38 | 68.38 |
| 32 | 44.29 | 25.48 | 21.67 | 18.00 | 18.17 | 43.07 | 25.88 | 28.31 | 47.29 |

**Table B.6:** Four mixture systems, SI and SX sentences - Recognition accuracy percentage as a function of the specified number of clusters and pruning threshold.

| Number of clusters | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 512 | 13.11 | 13.11 | 13.11 | 13.11 | 13.11 | 13.11 | 13.11 | 13.11 | 13.11 |
| 256 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 | 16.29 |
| 128 | 14.98 | 14.98 | 15.39 | 14.98 | 14.98 | 14.98 | 14.98 | 14.98 | 14.98 |
| 64 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 |
| 32 | 19.46 | 19.46 | 19.46 | 19.46 | 20.36 | 19.79 | 19.79 | 19.79 | 19.79 |

**Table B.7:** Two mixture systems, SA sentences - Recognition accuracy percentage as a function of the specified number of clusters and pruning threshold.

| Number of clusters | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 512 | 89.31 | 90.83 | 89.10 | 90.31 | 87.02 | 86.10 | 84.64 | 85.14 | 81.69 |
| 256 | 79.45 | 79.01 | 87.17 | 84.12 | 81.76 | 84.17 | 84.40 | 81.00 | 77.17 |
| 128 | 71.10 | 72.40 | 73.05 | 77.43 | 69.83 | 69.76 | 75.57 | 76.36 | 70.76 |
| 64 | 55.86 | 59.50 | 60.19 | 64.12 | 63.50 | 63.57 | 63.31 | 60.26 | 56.93 |
| 32 | 44.24 | 28.19 | 23.17 | 18.62 | 25.86 | 24.43 | 19.57 | 22.88 | 15.81 |

**Table B.8:** Two mixture systems, SI and SX sentences - Recognition accuracy percentage as a function of the specified number of clusters and pruning threshold.

| Number of clusters | Pruning threshold | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
| 512 | 13.44 | 13.44 | 13.44 | 13.44 | 13.44 | 13.44 | 13.44 | 13.44 | 13.44 |
| 256 | 16.12 | 16.12 | 16.12 | 16.12 | 16.12 | 16.94 | 16.12 | 16.12 | 16.12 |
| 128 | 15.39 | 15.39 | 15.39 | 15.39 | 15.39 | 15.39 | 15.39 | 15.39 | 15.39 |
| 64 | 21.25 | 21.25 | 21.25 | 21.42 | 21.25 | 21.25 | 21.25 | 21.25 | 21.25 |
| 32 | 19.54 | 19.54 | 19.54 | 19.54 | 19.54 | 19.54 | 19.54 | 19.54 | 19.54 |

# Appendix C

# Automatic segmentation results in graph format

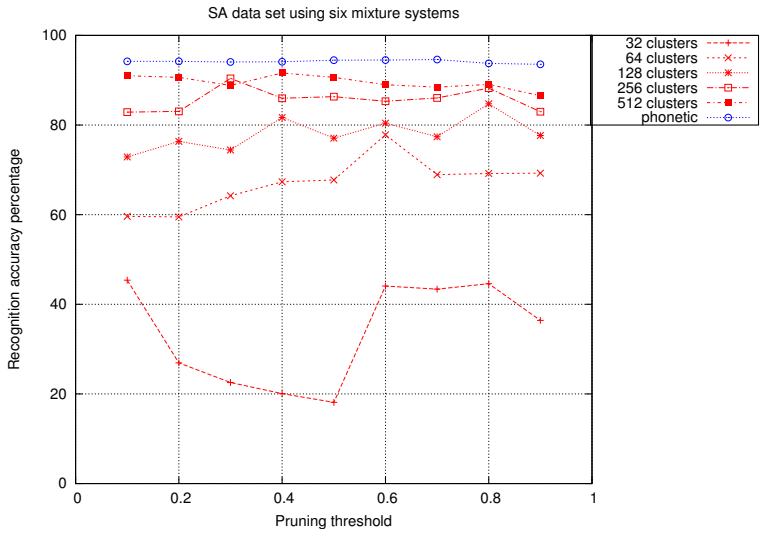



**Figure C.1:** Results of SA data set using six mixture systems.

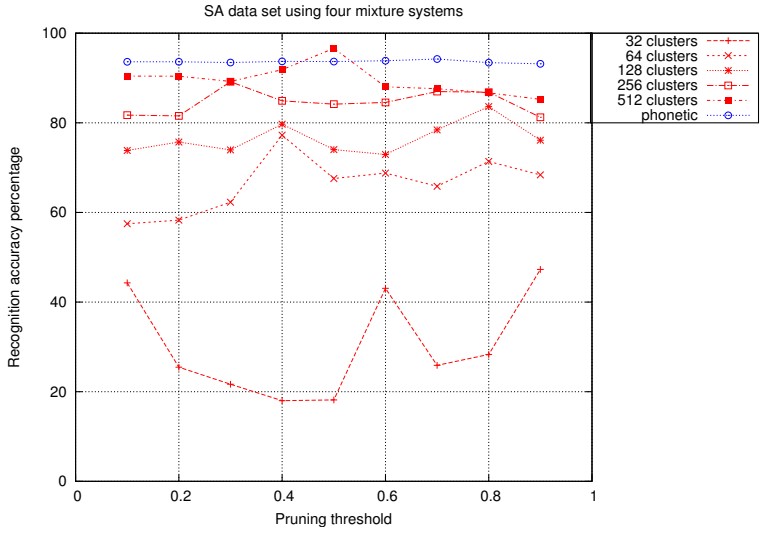



**Figure C.2:** Results of SA data set using four mixture systems.

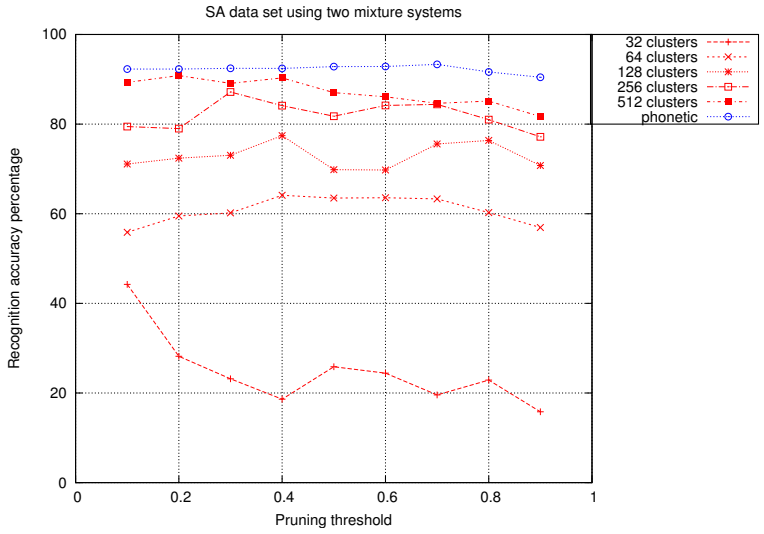



**Figure C.3:** Results of SA data set using two mixture systems.

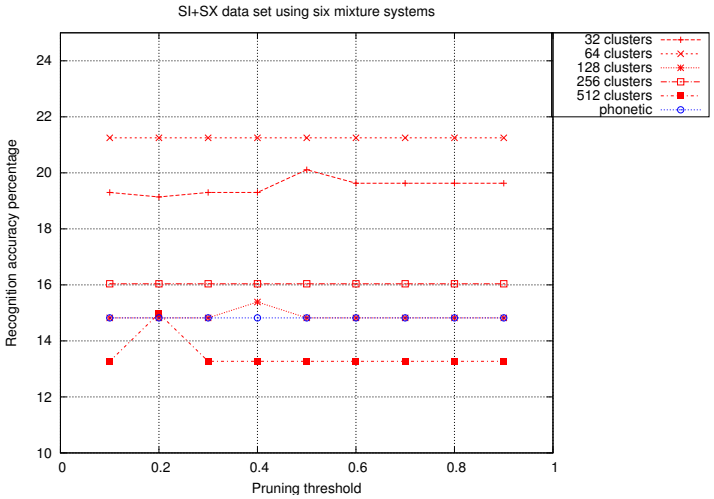



**Figure C.4:** Results of SI+SX data set using six mixture systems.

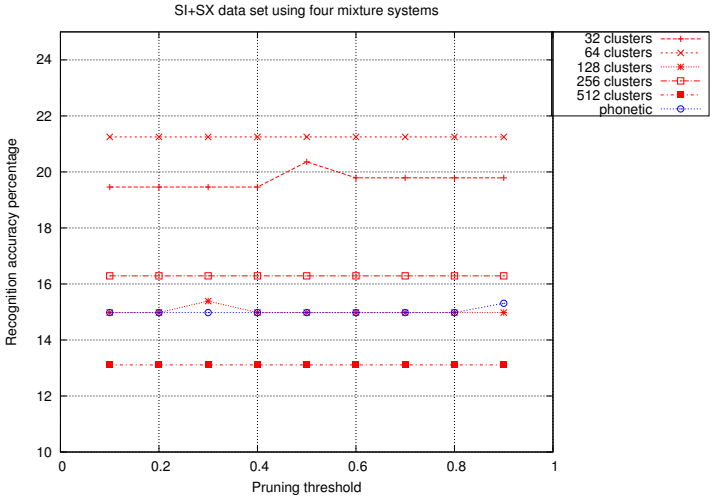



**Figure C.5:** Results of SI+SX data set using four mixture systems.

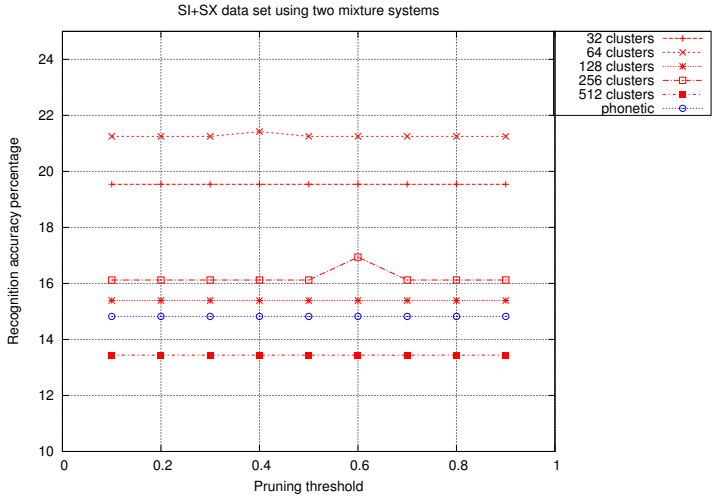



**Figure C.6:** Results of SI+SX data set using two mixture systems.